R. STANICIC

48

mi COMPUTER

CURSO PRACTICO DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

File Edit Goodies Font FontSize Style

Cat & Mouse



150 ptas.

Editorial Delta, S.A.

MI COMPUTER CURSO PRACTICO DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen IV - Fascículo 48

Director:	José Mas Godayol
Director editorial:	Gerardo Romero
Jefe de redacción:	Pablo Parra
Coordinación editorial:	Jaime Mardones
	Francisco Martín
Asesor técnico:	Ramón Cervelló

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Paseo de Gracia, 88, 5.°, 08008 Barcelona
Tels. (93) 215 10 32 / (93) 215 10 50 - Télex 97848 EDLTE

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 96 fascículos de aparición semanal, encuadernables en ocho volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-7598-005-8 (tomo 4)
84-85822-82-X (obra completa)
Depósito Legal: B. 52/1984

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 128412
Impreso en España - Printed in Spain - Diciembre 1984

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Argentina: Viscontea Distribuidora, S.C.A., La Rioja 1134/56, Buenos Aires.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (96 fascículos más las tapas, guardas y transferibles para la confección de los 8 volúmenes) son las siguientes:

a) Un pago único anticipado de 16 690 ptas. o bien 8 pagos trimestrales anticipados y consecutivos de 2 087 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S.A. (Paseo de Gracia, 88, 5.°, 08008 Barcelona), o también con talón bancario remitido a la misma dirección.

c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

Para cualquier aclaración, telefonear al (93) 215 75 21.

No se efectúan envíos contra reembolso.





¿Está usted siendo espiado?

KODAK SAFETY FILM 5017

→36 →36A 5A

Steve Cross

Espionaje en el despacho
Existen muchísimos lugares donde se pueden ocultar dispositivos de escucha, desde los más obvios hasta los más insólitos. En este despacho de ejecutivo, los siguientes lugares podrían estar ocultando "micrófonos":

1) El teléfono. El micrófono podría estar en la bocina o el cuerpo del teléfono, o bien en la lámpara, que está muy cerca.

2) Planta en maceta. En la tierra, debajo de la maceta, ¡e incluso simulando un insecto!

3) La pared. Se sabe que se han empotrado micrófonos en la pared a modo de "clavos" o detrás de paneles de corcho.

4) Cuadro colgado. Disimulado como gancho de apoyo u oculto detrás.

5) En el escritorio. Debajo de la tabla superior o en cualquiera de los cajones

En este capítulo pasamos revista a los últimos adelantos en técnicas de vigilancia informatizada

El empleo oficial de la vigilancia por ordenador se está ampliando de forma continua, introduciéndose en cada vez más aspectos de la vida cotidiana. Con certeza todos hemos sido incluidos en alguno de estos sistemas en algún momento de nuestra vida.

En Gran Bretaña, por ejemplo, país bastante avanzado en este aspecto, existen vigilancias que parecen inocentes y que revisten vital importancia para las agencias del gobierno, como los registros de automóviles y matrículas de los ordenadores DVLC de Swansea o los registros de los DHSS basados en ordenador que pertenecen a la Seguridad Social. Ahora se ha planteado la posibilidad de enlazar todos estos distintos sistemas entre sí, para correlacionar los archivos de los sistemas DVLC y DHSS con, pongamos por caso, los archivos del ordenador de la Policía Nacional.

La Data Protection Act (Ley para la Protección de Datos) se formuló en un intento por detener los abusos de este tipo de poder. Pero hay quienes creen que la misma está anticuada, a la luz de los rápidos avances que se han producido en la tecnología del ordenador desde que fuera introducida.

Muchas de las técnicas que se utilizan en la vigilancia por ordenador y los sistemas de seguridad implican el reconocimiento de patrones, una técnica en virtud de la cual el ordenador compara lo que "ve" con patrones que ya tiene almacenados en su memoria. El inconveniente del reconocimiento de patrones es que exige grandes cantidades de espacio de memoria y enormes cantidades de capacidad de proceso del ordenador. En la actualidad se dispone de ambas cosas y a un precio reducido, lo que ha facilitado la realización de significativos avances.

Un ejemplo de ello es el nuevo sistema de impresión dactiloscópica que ha instalado Logica para la Policía Metropolitana en la Nueva Scotland Yard de Londres. Han sido necesarios 15 años de desarrollo para crear el sistema, que puede almacenar 650 000 impresiones dactilares y 100 000 "huellas" (las impresiones parciales que se encuentran en el escenario de un delito). El sistema simplemente compara las huellas con todas las impresiones almacenadas, para ver si concuerdan con alguna de las del archivo. Esta aplicación necesita de la potencia de cálculo de miniordenadores Prime, junto con procesadores matriciales sumamente eficientes y

Herramientas de la profesión



Protección informatizada
Muchos ejecutivos y personas que trabajan con información clasificada se protegen a sí mismos contra el espionaje electrónico con artilugios como este mezclador telefónico informatizado. En vez de hablar por el teléfono, el usuario digita sus mensajes en el teclado. El mezclador envía el mensaje en forma "silenciosa" a través de las líneas telefónicas normales. El mensaje se recibe entonces en la pantalla incorporada del sistema. Un sintetizador de voz incorporado también puede convertir el mensaje que se recibe a una forma verbal audible con sólo tocar un botón



Mensaje cifrado
Parecido al mezclador informatizado, este sistema envía un mensaje mezclado a partir de una nota manuscrita. Éste se encuentra protegido contra escuchas furtivos y micrófonos ocultos porque se envía silenciosamente. De esta forma se pueden enviar hasta firmas



Analizador de tensión
Este analizador mide la tensión de la voz y visualiza sus resultados instantáneamente en una sencilla lectura numérica. Se trata en realidad de un sofisticado detector de mentiras y también puede indicar si la persona experimenta ansiedad o se halla en tensión

pantallas y cámaras de televisión de elevado rendimiento. Aun así, sólo puede comparar cada día 200 o 300 huellas con las 650 000 impresiones almacenadas.

Un sistema similar de reconocimiento y comparación de patrones está instalado en un puente que atraviesa la autopista M1, con cámaras que enfocan hacia cada uno de los carriles. En realidad este sistema capta imágenes del número de matrícula de los coches cuando éstos se van aproximando y luego utiliza la potencia de cálculo del ordenador para analizar las imágenes y comparar los números con un archivo de vehículos buscados. La información de que se ha visto uno de los vehículos buscados se puede entonces transmitir por radio directamente a los coches patrulla de la autopista, que lo interceptarán.

Uno de los campos obvios en el que los desarrollos en el hardware para ordenador tuvieron una trascendental influencia fue la producción de dispositivos para vigilancia cada vez más pequeños; es decir, en los micrófonos ocultos. La tecnología del chip ha hecho posible fabricar transmisores de radio del tamaño de un grano de arroz, con una sofisticada electrónica de control incorporada. Un dispositivo típico "chupa" su potencia desde la fuente de alimentación eléctrica de la Central Telefónica hasta el teléfono intervenido, y sólo se conecta a sí mismo cuando efectivamente hay alguien hablando. Luego están los micrófonos de autoalimentación, igualmente diminutos, que se dejan caer en un rincón de una habitación y recogen todas las conversaciones que tienen lugar en ella (al igual que los arriba citados, sólo entran en funcionamiento cuando alguien habla) para transmitir a un receptor remoto.

Más próximo aún al "estilo James Bond", existe un micrófono "a distancia" que dispara un haz de láser hacia una ventana. Las vibraciones que la conversación produce en el cristal se captan como interferencia en el haz de láser reflejado y la información verbal se extrae de esta interferencia (por ordenador, por supuesto).

En las operaciones militares, al contrario que en las tareas de seguridad clandestinas, los operadores de ordenadores se encuentran con el problema opuesto. Intentan evitar ser controlados por otras personas y, ¡faltaría más!, son los chips y los ordenadores los que han venido a auxiliarlos. Los transmisores y receptores de radio actuales para el campo de batalla utilizan el salto de frecuencia (saltar, bajo el control del procesador, de una frecuencia a otra según un código preestablecido) para evitar escuchas secretas y la perturbación con señales de interferencia.

Los ordenadores para vigilancia y seguridad son, en la actualidad, grandes máquinas del tipo y la potencia de las que se emplean para la decodificación de claves complejas en lugares como la CIA y la National Security Agency, en Estados Unidos, por no mencionar el M15 y el M16 de Gran Bretaña. Pero los avances en la tecnología del hardware hacen presumir que pronto el reconocimiento de impresiones dactiloscópicas (y tal vez el de rostros) se automatizará y resultará muy económico.

En el futuro, es probable que los coches de la policía se equipen con ordenadores a bordo que puedan extraer datos instantáneamente de un registro escolar, un registro criminal, un registro médico, de la Seguridad Social o cualquier otro archivo oficial. A todos ellos se accedería introduciendo en la ranura del ordenador una tarjeta plástica de la Seguridad Nacional. La máquina aceptaría huellas dactilares y una fotografía, las compararía con los archivos centrales y confirmaría la identidad del sospechoso.

Un suave toque

Dominar la mecanografía al tacto es esencial para muchos aspectos de la informática personal

La capacidad de teclear palabras enteras sobre el papel o para contemplarlas en una pantalla, en vez de ir pulsando laboriosamente letras individuales en un teclado, elimina un paso que ocupa mucho tiempo en un proceso en el cual la velocidad suele tener importancia, y reduce el margen de error.

Tradicionalmente la mecanografía se ha enseñado mediante manuales de instrucción o en la clase, utilizando diversos métodos. Existe incluso en el mercado una gama de programas de software que conecta las lecciones directamente al ordenador. Independientemente del método de enseñanza, hay cuatro principios que se deben aprender para convertirse en un consumado mecanógrafo al tacto. Éstos son:

- Dominar el teclado
- Acostumbrarse a fijar los ojos en una pantalla (o papel)
- Precisión
- Velocidad

Estas habilidades se desarrollan de modo lento, mediante ejercicios, hasta que el mecanógrafo alcanza un nivel de destreza. La repetición constante es el principal método de aprendizaje: golpear las mismas letras hasta que la secuencia de los dedos se vuelve automática.

La disposición estándar en lengua inglesa para una máquina de escribir se conoce como teclado QWERTY, así llamado porque ésas son las letras situadas al lado izquierdo de la segunda de las cuatro filas de teclas. Los símbolos y las letras del teclado están dispuestos de acuerdo a su frecuencia de uso y cada fila del teclado, al objeto de la enseñanza, está dividida en lado izquierdo y lado derecho.

El aprendizaje de la mecanografía al tacto empieza por lo general por el dominio de las ocho teclas de la fila de letras del medio. Éstas se conocen como *letras base*, porque los dedos retornan a ellas después de haber pulsado cualquier otra tecla del teclado. Las teclas base para el lado izquierdo de la máquina de escribir y, por consiguiente, los cuatro dedos de la mano izquierda, son asdf. Las teclas base para el lado derecho de la máquina de escribir y, por tanto, para la mano derecha, son jkl; (punto y coma). Después de dominar estas teclas, el mecanógrafo aprende la situación de las otras en relación a su posición respecto a las teclas base, palpándolas y desplazándose hacia ellas estando aún en la posición "base". Una sorpresa con la que se encuentra el recién iniciado en la mecanografía es el uso al que se destina el dedo meñique. La mecanografía al tacto exige la utilización de *todos* los dedos y el meñique tiene asignada una cantidad de teclas "a cubrir", al igual que los demás dedos. Después de aprenderse las teclas, el mecanógrafo pasa a la barra espaciadora (que se pulsa con los pulgares) y a la tecla de cambio para las letras mayúsculas y las teclas que tienen asociados dos caracteres.

Lamentablemente para el usuario de un ordena-



Tony Sleep

Vista y oído
Sight and Sound ofrece cursos de mecanografía al tacto para principiantes y ha dado buenas pruebas de éxito. En esta fotografía, se les está enseñando a los estudiantes mediante una combinación de técnicas visuales y auditivas

dor personal, la utilización correcta de las teclas numéricas es un tema que se trata muy de vez en cuando (por no decir nunca) en los cursos o manuales de mecanografía al tacto. Para el programador, en especial, las teclas numéricas constituyen una parte vital e integral del teclado del ordenador. La mecanografía de números al tacto es bastante fácil de aprender una vez dominados el concepto y el funcionamiento de las teclas base. Los dedos de la mano izquierda simplemente se estiran por encima de las teclas alfabéticas para cubrir los números del uno al seis, mientras que los dedos de la mano derecha son responsables desde el siete al cero y de las teclas de signos que haya a continuación.

Se puede efectuar una interesante extensión del enfoque de la "tecla base" cuando se utiliza un ordenador personal con caracteres para gráficos a los que se puede acceder directamente desde el teclado. Aprendiendo la situación de los símbolos de gráficos, sería posible incorporarlos a su programa de mecanografía al tacto.

A fijar los ojos en una pantalla (o en un papel) se aprende al mismo tiempo que se va dominando el teclado. Éste es el aspecto más importante de la mecanografía al tacto, lo que la distingue del "teclado a dos dedos" o de mecanografiar "mirando y buscando". Tapar las teclas con cinta adhesiva o con unas cubiertas diseñadas especialmente es una excelente ayuda para el principiante, ya que impide caer en la tentación de bajar la vista hacia las manos. En este sentido, un teclado y una pantalla son mejores que una máquina de escribir y una hoja de papel. La pantalla está a la altura de los ojos, por lo cual disminuye la tentación de mirar hacia abajo. También se pueden corregir de inmediato los errores, asegurando mayor precisión, que es la siguiente etapa de la mecanografía al tacto.

La precisión es cuestión de práctica y concentración. Debe pulsarse la tecla con un golpecito firme y rápido, directamente en el centro, tocando las teclas con un mismo grado de regularidad, lo que produce un ritmo que, llegado el momento, se vuelve natural. Finalmente, utilizar el dedo equivocado o pulsar la tecla errónea llega a resultar extraño, y este ritmo natural contribuye en gran medida a adquirir velocidad.

La velocidad se mide mediante pruebas de tiempo y ejercicios, y se obtiene sólo después de haber superado las tres etapas que hemos descrito anteriormente. Los mejores mecanógrafos al tacto pueden alcanzar velocidades que superan las 100 palabras por minuto (ppm). Para el principiante, una velocidad razonable para plantearse como objetivo es de alrededor de 30 ppm.

Existen varios manuales de enseñanza diferentes y en el transcurso de los años se han ido desarrollando numerosos métodos. Aun así, todos ellos siguen valiéndose de los principios que hemos explicado. Pitman, la escuela internacional de secretariado, todavía edita un manual que se publicó por primera vez hace 35 años, así como una versión actualizada que enseña el teclado a través del mecanografiado de *palabras* desde el primer ejercicio. Dado que en estos ejercicios iniciales se utilizan palabras cortas y de uso común, la ortografía no exige ningún esfuerzo, lo que deja al principiante libre para concentrarse en adquirir la técnica necesaria.

El Dico Typing Course se precia de haberle ense-

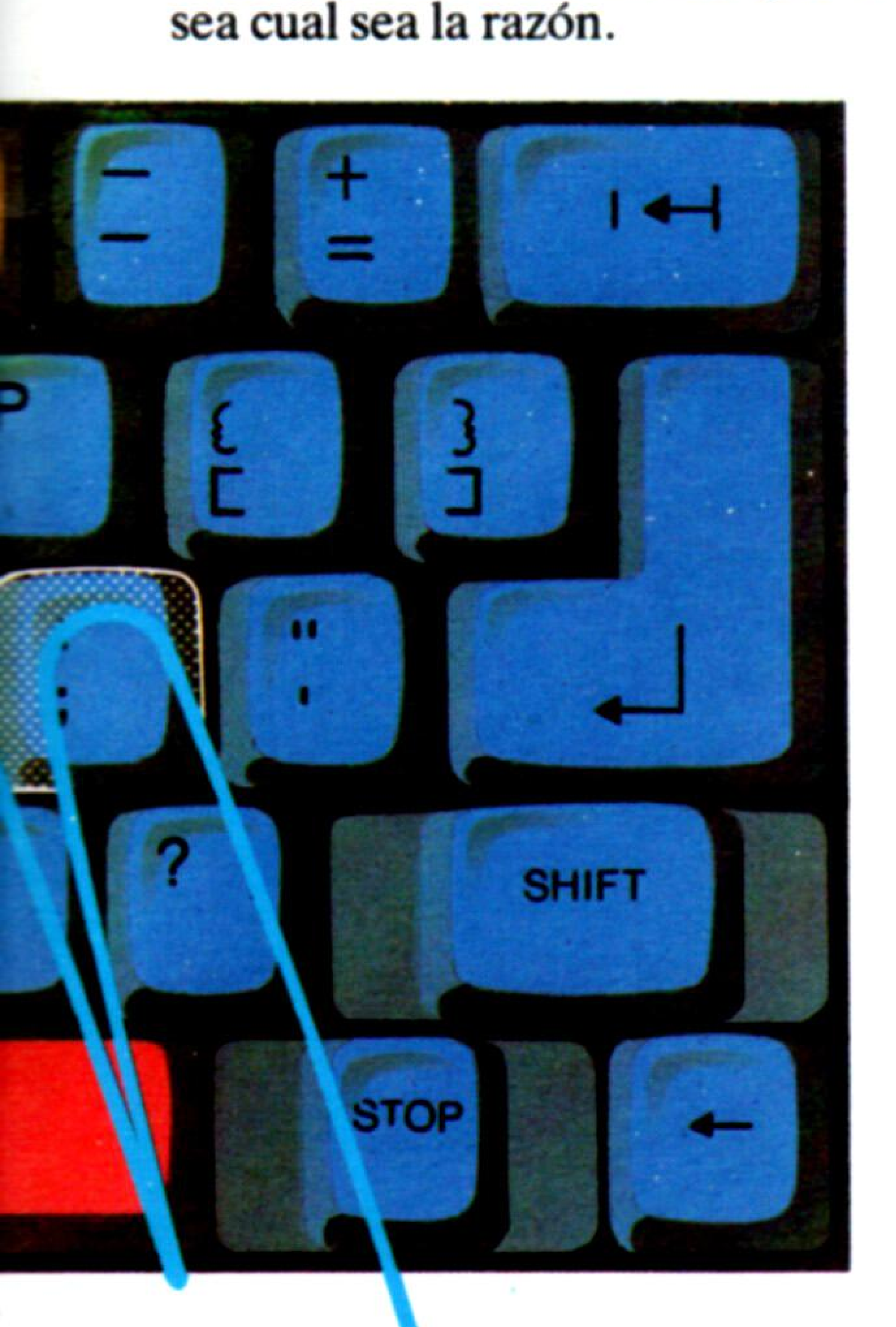


Base de operaciones
Esta ilustración muestra la situación de las teclas base para las manos izquierda y derecha y las teclas que corresponden a cada dedo. Los dedos índice y meñique, por ser los que tienen más libertad de movimiento, son los encargados de pulsar el mayor número de teclas

Kevin Jones

ñado mecanografía al tacto a niños de 11 años de edad en menos de 10 horas. Este enfoque se basa en la obra del psicólogo norteamericano BF Skinner. En este método, el principiante arranca la página del manual y mecanografía la respuesta debajo del ejercicio. Esto responde al tradicional método de enseñanza según el cual el alumno copia exactamente ejercicios preparados. Los ejercicios se acompañan con dibujos de manos colocadas en la posición correcta sobre el teclado.

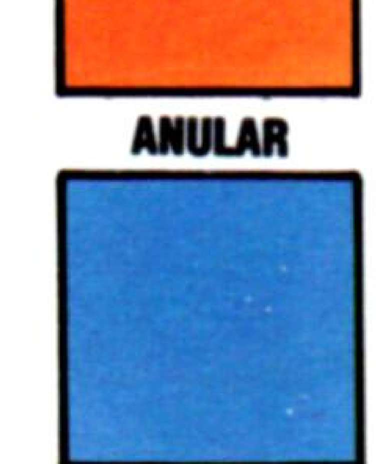
Una escuela que pone los métodos de mecanografía al tacto directamente en línea con el pensamiento moderno es Sight and Sound. Esta institución posee 11 centros de adiestramiento en Gran Bretaña y otros países. Los métodos de enseñanza de Sight and Sound implican la utilización de luces intermitentes y cassettes grabadas. Este sistema de enseñanza audiovisual simula las respuestas de ver, oír y reaccionar simultáneamente. En un gran tablero elevado una luz intermitente ilumina una letra. La cinta grabada se sincroniza con el tablero y la voz pregrabada del instructor al exclamar "¡Ya!" determina la velocidad a la cual se han de mecanografiar las letras. A medida que uno va adquiriendo más experiencia, se incrementa la velocidad a la cual el instructor exige que se mecanografíen las letras. La técnica de Sight and Sound se ha descrito como "lavado de cerebro", e incluso los instructores admiten que no comprenden del todo por qué este método produce resultados tan eficaces. No obstante, la cantidad de alumnos satisfechos que aprenden a mecanografiar al tacto con eficacia y sin esfuerzo indica que el sistema funciona, sea cual sea la razón.





ÍNDICE

CORAZÓN



ANULAR

MEÑIQUE

Sumario del software

Los programas para ordenador que enseñan mecanografía al tacto se basan en juegos o en texto. Los paquetes basados en texto se fundamentan sobre todo en ejercicios y repetición de procedimientos. La mayor parte de las lecciones de los programas de esta clase se basan en texto, por lo general en forma de ejercicios escritos que el usuario debe copiar exactamente. Los paquetes basados en juegos enseñan a través de la utilización de gráficos, acción rápida y sonido



Ian McKinnell

Type Invaders

Editado por: Carswell Computers
Máquina: BBC Modelo B

Éste es un paquete basado en un juego para quienes poseen nociones de mecanografía. Sus sencillos gráficos visualizan letras "atacantes" que se deben destruir mecanografiándolas correctamente. Las palabras y las letras que se han fallado vuelven a atacar y finalmente pueden acabar rompiendo sus líneas de defensa y ocupando su territorio. Hay 10 niveles diferentes de juego, que van desde letras mayúsculas solamente hasta palabras de cinco letras que contienen mayúsculas y minúsculas, mayúsculas y cifras. También hay tres velocidades para escoger: fácil, rápida y veloz. A los mecanógrafos experimentados hasta la opción "veloz" les resultará sencilla. Sin embargo, se trata de un paquete práctico y entretenido para mecanógrafos lentos y normales. Los principiantes deberían practicar primero con el paquete Typeasy, del mismo fabricante

Sprintyper

Editado por: Micro Software International
Máquina: Commodore Vic-20

Paquete basado en texto que afirma mejorar la velocidad y precisión tanto de principiantes como de mecanógrafos avanzados. Posee una biblioteca de 356 635 frases para mejorar la habilidad. Para comenzar, se muestra en la pantalla una frase sencilla que se debe mecanografiar lo más rápidamente posible. Un tono grave señala un error, y sólo cesa al efectuar la corrección. Después de copiar la frase correctamente, aparecen en la pantalla el tiempo de mecanografiado, la cantidad de errores y un tiempo récord. Sprintyper es, esencialmente, una prueba de velocidad, que tiene poco que ofrecer al principiante en cuanto a ejercicios constructivos

Typing Tutor II

Editado por: Microsoft
Máquinas: Apple IIe y Apple IIe+

Para ejecutar este paquete se necesita Applesoft en ROM, 48 K de memoria, una unidad de disco y el sistema DOS 3.3. Éste es un paquete basado en texto conducido por menú, que proporciona una combinación de lecciones, párrafos de práctica y pruebas de velocidad. Su característica más importante es el sistema *Time response monitoring* (Control de tiempo de respuesta), que comprueba el mecanografiado 100 veces por segundo, detectando hasta las pausas más imperceptibles que se producen si los ojos se desvían de la pantalla al teclado. Los principiantes comienzan con cierta cantidad de letras para practicar. Cuando se van familiarizando con estas letras, y cuando la velocidad de la mecanografía equivale a 30 palabras por minuto, esas letras se transfieren a una columna FAST (rápido) y se seleccionan nuevas letras para practicar. Para los mecanógrafos experimentados, el informe de progreso del párrafo de práctica detalla la cantidad de errores cometidos, las teclas con las que se incurrió en esos errores, la velocidad y la precisión. Es un paquete muy aconsejable para los mecanógrafos de cualquier nivel de destreza. Al principio resulta un poco difícil de seguir y es recomendable que los usuarios previamente estudien con sumo cuidado la documentación

Brillo y esplendor

Ahora añadiremos al programa refinamientos que harán el juego más atractivo y emocionante

El primer añadido que efectuaremos es una rutina de "francotiroteo". Ésta simula a un francotirador que dispara hacia el campo de minas tratando de hacer blanco en el detector de minas o bien en el ayudante. Los disparos se visualizarán como una línea en alta resolución que atravesará la pantalla desde el margen izquierdo del campo de minas hacia el derecho. Para introducir en el francotiroteo un elemento de azar, seleccionaremos las coordenadas de los puntos de comienzo y final utilizando la función RND. Los valores de xcomienzo y xfinal se establecen en el procedimiento inicializarvariables. La diferencia entre estos dos valores es de 1 024 unidades de gráficos. Para que la línea de francotiroteo pueda detectar un blanco, ya sea en el detector o en el ayudante, ha de trazar un corto segmento de la línea y verificar la superficie que hay por delante en busca de la presencia del color lógico 1 (utilizando la instrucción POINT) antes de trazar el segmento siguiente. Esta secuencia se debe repetir hasta alcanzar el otro lado de la pantalla o hacer blanco.

Ahora debemos decidir qué longitud deseamos utilizar para los pasos. Si elegimos una longitud de paso muy corta, entonces aumentará el tiempo que se tarda en trazar la línea. Por el contrario, si elegimos una longitud de paso demasiado larga quizá no consigamos detectar los objetivos. Dado que cada celda de caracteres equivale a lo ancho a 64 unidades para gráficos, parece razonable una longitud de paso de media celda de caracteres (es decir, 32 unidades para gráficos). Por consiguiente, si optamos por que nuestra longitud de paso en la dirección *x* (dx) sea de 32 unidades, podemos trazar la línea en un total de 1024/32 = 32 pasos. Si calculamos las coordenadas *y* de los puntos de comienzo y final al azar, entonces la longitud de paso adecuada en la dirección *y* (dx) se puede calcular dividiendo por 32 la diferencia entre los dos valores.

Nuestro último problema es hallar alguna forma de borrar la línea después de haberla trazado. La solución radica en el concepto de colores lógicos del BASIC BBC y su capacidad para realizar operaciones lógicas entre ellos. En la modalidad 5 hay cuatro colores lógicos. A menos que los modifiquemos, éstos son:

Color lógico	0	1	2	3
Equivalente binario	00	01	10	11
Color real normal	negro	rojo	amarillo	blanco

Utilizando GCOL podemos efectuar varias operaciones lógicas entre el color que estamos trazando y el color que ya está allí. La instrucción posee dos parámetros, el segundo de los cuales indica el color lógico a trazar. El primer número establece el método de trazado:

GCOL0	Traza el color especificado
GCOL1	Realiza operación OR
GCOL2	Realiza operación AND
GCOL3	Realiza operación OR Exclusivo
GCOL4	Realiza NOT en el color que ya está allí

Esto puede parecer complicado, pero con unos pocos ejemplos clarificaremos el funcionamiento de la instrucción. Si está presente el blanco (color lógico 3) en la posición que deseamos trazar, y queremos trazar en rojo (color lógico 1), las diversas modalidades de operación de GCOL producirán los siguientes resultados:

GCOL0,1	Borrará el blanco y trazará en rojo		
GCOL1,1	Operará rojo y blanco con OR para producir blanco	rojo blanco OR blanco	01 11 11
GCOL2,1	Operará rojo y blanco con AND para producir rojo	rojo blanco AND rojo	01 11 01
GCOL3,1	Operará rojo y blanco con OR Exclusivo para producir amarillo	rojo blanco amarillo	01 11 10
GCOL4,1	Operará blanco con NOT para producir negro	blanco negro	11 00

¿Y cómo puede esto ayudarnos a solucionar nuestro problema de borrado? Podríamos trazar la línea en blanco y después volver a trazarla en negro para borrarla. Pero si debajo de la línea hubiera algo, como por ejemplo una mina, entonces esto haría que quedara un "agujero". Sin embargo, podemos operar con OR Exclusivo el rojo y el color ya presente en cada punto que atraviesa la línea. Cuando atraviese un área blanca, obtendremos un segmento de línea amarillo. Si operamos después la misma zona con OR Exclusivo en rojo el resultado sería:

rojo	01
amarillo	10
EOR	—
blanco	11

Por consiguiente, se retoma el color original. Quizá desee comprobar que efectuar dos OR Exclusivos siempre restaura el color original. Podemos emplear este factor para borrar nuestra línea. Si trazamos la línea original utilizando una operación EOR y después volvemos a trazar exactamente la misma línea, utilizando otra vez OR Exclusivo, borraremos la línea y restauraremos los colores de fondo a su condición original antes del primer trazado. He aquí el listado completo para el procedimiento francotirador:

```
3110 DEF PROCfrancotirador
3120 ycomienzo = RND(750) + 220
3130 yfinal = RND(750) + 220
3140 dx = 32:dy = (yfinal - ycomienzo)/32
3150 GCOL 3,3
3160 PROClinea
3170 IF POINT(x,y) = 1 THEN PROCexplotar(x,y) ELSE PROClinea
3180 ENPROC
```

Y éste es el listado para el procedimiento linea:

```
3450 DEF PROClinea
3460 SOUND0,-8,4,5
3470 x = xcomienzo:y = ycomienzo
3480 MOVE x,y
3490 REPEAT
3500 DRAW x,y
3510 x = x + dx:y = y + dy
3520 UNTIL x > xfinal OR POINT(x,y) = 1
3530 ENDPROC
```

Como vimos en el último capítulo, el BBC Micro puede generar sonidos bastantes complejos. Para quienes sientan inclinaciones musicales, ahora vamos a añadir al programa una corta melodía. Para hacer las cosas lo más sencillas posible utilizaremos sólo un canal. La melodía se puede ejecutar simplemente especificando la frecuencia y la duración de cada una de sus notas.

```
4090 DEF PROCmusica
4100 REM ** 1A BARRA **
4110 SOUND1,-8,213,5
4120 SOUND1,-8,209,5
4130 SOUND1,-8,213,5
4140 SOUND1,-8,209,5
4150 SOUND1,-8,213,5
4160 SOUND1,-8,193,5
4170 SOUND1,-8,205,5
4180 SOUND1,-8,197,5
4190 REM ** 2A BARRA **
4200 SOUND1,-8,185,20
4210 SOUND1,-8,165,5
4220 SOUND1,-8,185,5
4230 SOUND1,-8,193,20
4240 REM ** 3A BARRA **
4250 SOUND1,-8,165,5
4260 SOUND1,-8,193,5
4270 SOUND1,-8,197,20
4280 ENDPROC
```

Página de títulos: Podemos utilizar las ideas del trazado con OR Exclusivo y el trazado de puntos relativos para crear una interesante secuencia de títulos. Este procedimiento dibuja la palabra MINES (minas) utilizando gráficos en alta resolución. Cada nueva línea dibujada en la palabra se traza en relación a la última, de modo que podemos situar la palabra completa en cualquier lugar de la pantalla simplemente especificando el punto de comienzo. Si trazamos la palabra y luego la volvemos a trazar en OR Exclusivo antes de desplazarla hacia arriba y repitiendo la acción, podemos lograr que la palabra parezca que flote hacia arriba de la pantalla. GCOL0,129 establece el color del fondo en rojo. Efectuando un subsiguiente CLG toda la pantalla se colorea de rojo. Al mismo tiempo, también podemos tocar la melodía definida anteriormente llamando a PROCmusica. La información retenida en PROCmusica se procesa bastante más rápidamente que lo que se toca, de modo que se utiliza un buffer para almacenar la información SOUND hasta que es tocada. Ello significa que el procesador queda libre para realizar otras cosas mientras todavía está sonando la melodía.

Factores de destreza: Para hacer que el juego resulte un poco más emocionante podemos emplear la noción de factores de destreza. Después de haberse visualizado el título, solicitaremos un número entre 0 y 9, que se almacenará en la variable destreza. Éste se puede utilizar entonces para aumentar la cantidad de minas en el campo de minas y la velocidad a la cual el francotirador dispara sobre la zona. Lo primero se puede realizar introduciendo una ligera modificación en el procedimiento preparación que dimos con anterioridad (véase p. 885). Cambie las líneas 1930 y 1940 por:

```
1930 factor = destreza*3 + 30
1940 PROCcolocar-minas(factor)
```

Además, cuando volvemos a colocar las minas durante el procedimiento restaurar, debemos calcular la cantidad de minas que queda modificando de este modo la línea 3950:

```
3950 minas-restantes = factor-marcador/150
```

El listado del procedimiento pagina de titulos es:

```
1300 DEF PROCpagina-titulos
1310 GCOL 0,129
1320 CLG
1330 GCOL 3,3
1340 PROCmusica
1350 Y = 100:X = 0
1360 REPEAT
1370 X = X + 20:Y = Y + 50
1380 FOR I = 1 TO 2
1390 PROCminas
1400 NEXT I
1410 UNTIL Y > 700
1420 :
1430 PROCminas
1440 PRINTTAB(0,20) "Factor destreza (0-9)?"
1450 PROCmusica
1460 REPEAT
1470 destreza = GET-48
1480 UNTIL destreza >-1 AND destreza < 10
1490 ENDPROC
1500 :
1510 DEF PROCminas
1520 PLOT4,X,Y
1530 REM ** LETRA M **
1540 PLOT1,0,200
1550 PLOT1,80,-100
1560 PLOT1,80,100
1570 PLOT1,0,-200
1580 REM ** LETRA I **
1590 PLOT0,40,0
1600 PLOT1,80,0
1610 PLOT0,-40,0
1620 PLOT1,0,200
1630 PLOT0,-40,0
1640 PLOT1,80,0
1650 REM ** LETRA N **
1660 PLOT0,40,-200
1670 PLOT1,0,200
1680 PLOT1,120,-200
1690 PLOT1,0,200
1700 REM ** LETRA E **
1710 PLOT0,160,0
1720 PLOT1,-120,0
1730 PLOT1,0,-200
1740 PLOT1,120,0
1750 PLOT0,-40,100
1760 PLOT1,-80,0
1770 REM ** LETRA S **
1780 PLOT0,280,60
1790 PLOT1,0,40
1800 PLOT1,-120,0
1810 PLOT1,0,-100
1820 PLOT1,120,0
1830 PLOT1,0,-100
1840 PLOT1,-120,0
1850 PLOT1,0,40
1860 ENDPROC
```

Hasta este punto hemos estado utilizando un programa de llamada provisional (véase p. 874) para ensamblar nuestros procedimientos entre sí, pero ahora hemos ensamblado todos los procedimientos que se requieren para el bucle del programa principal del juego. Borre el programa de llamada provisional (líneas 10 a 70) e incorpore lo siguiente:

```
2020 DEF PROCbucle
2030 REPEAT
2040 PROCactualizar-tiempo
2050 PROCleer-teclado
2060 rand = RND(50-destreza)
2070 IF rand = 1 THEN PROCfrancotirador
2080 UNTIL TIME > 12099 OR flag-final = 1
2090 ENDPROC
```

Ahora podemos escribir nuestro programa de llamada. Entre estas líneas:

```
1060 max-marcador$ = "00000"
1110 MODE5
1120 REM ** APAGAR CURSOR **
1130 VDU23;8202;0;0;0;
1140 PROCpagina-titulos
1150 CLS
1160 PROCpreparacion
1170 :
1180 PROCbucle
```

Corte de secuencia

Un proceso general puede representarse dividido en varias partes gracias a la utilización de un símbolo especial

El símbolo // se emplea en diagramación para representar un corte en la secuencia; siempre que aparezca al principio o al final de una representación se entenderá que lo que figura forma parte de un proceso general más amplio. En el ejemplo siguiente se han realizado incisiones en el ordinograma general, diseccionándolo en tres cuerpos.

Una máquina colocada al final de una línea de producción controla los tres tipos de piezas que se fabrican (A, B y C). Su función es pesar cada una de las piezas que, filtradas, llegan a ella e imprimir su peso. Así sucesivamente hasta que finaliza la producción diaria; entonces debe dar los pesos totales de las piezas de cada tipo y el total general.

Haciendo un análisis del problema, llegamos a la conclusión de que hay una parte común para todas las piezas, independientemente del tipo al que pertenezcan, que es la correspondiente a la entrada y pesaje, así como la impresión del peso (fig. 1).

Nuestro problema termina con la pregunta de si quedan más piezas por procesar. En caso afirmativo se retorna al principio del bucle; en caso negativo se llega a la salida de los datos que se precisan recoger como final de jornada, a saber, tres totales parciales y un total general. También en este caso la obtención del peso total general se lleva a cabo mediante la suma de los tres acumuladores parciales, sin tener que recurrir innecesariamente al empleo de un acumulador general (fig. 3).

Pero para obtener estos resultados antes se ha debido llevar a cabo una serie de pasos que vienen marcados por una sucesión de preguntas eliminatorias *(test en cascada)* destinadas a conocer, a través de su peso, el tipo de pieza que se ha controlado, para así sumar su peso al correspondiente acumulador (fig. 2).

Figura 1
INICIO
TOMAR PIEZA
PESARLA
PESO

Figura 3
TIPO A ?
SÍ
SUMAR PESO AL ACUM. A
NO
TIPO B ?
SÍ
SUMAR PESO AL ACUM. B
NO
SUMAR PESO AL ACUM. C

Figura 2
MÁS PIEZAS ?
NO
ACUMULADOR A
ACUMULADOR B
ACUMULADOR C
ACUM. A + ACUM. B + ACUM. C
FIN
A

Ordinograma general
INICIO
A
TOMAR PIEZA
PESARLA
PESO
TIPO A ?
SÍ
SUMAR PESO AL ACUM. A
NO
TIPO B ?
SÍ
SUMAR PESO AL ACUM. B
NO
SUMAR PESO AL ACUM. C
MÁS PIEZAS ?
SÍ
A
NO
ACUMULADOR A
ACUMULADOR B
ACUMULADOR C
ACUM. A + ACUM. B + ACUM. C
FIN



Precursor del futuro

El elegante e innovador Macintosh de Apple ha marcado un hito definitivo en el diseño de microordenadores

El Macintosh es distinto de todos los otros ordenadores que hemos analizado hasta ahora. En realidad, es diferente de cualquier otra máquina que existe en el mercado. A pesar de que el Macintosh es principalmente una máquina de oficina, Apple ha optado por crear su propio camino en vez de imitar a otros fabricantes que han adoptado los estándares IBM para los diseños de sus máquinas. Tomando este arriesgado camino, Apple ha mantenido su reputación de innovador en una industria llena de "semejanzas".

La disposición de los elementos del Macintosh es inusual. La elegante y esbelta unidad del sistema es pequeña para una máquina de su capacidad de proceso. La visualización se realiza en una pantalla de nueve pulgadas de alta resolución, y la unidad de disco utiliza discos Sony de 3 ½ pulgadas. En la carcasa se ha incorporado un asa moldeada para transportarlo, de modo que el Macintosh se puede considerar como una auténtica máquina portátil. Junto con el teclado, el "ratón" y un maletín opcional para su transporte, el sistema pesa en total 11,6 k. El maletín tiene compartimientos para todos los componentes del ordenador, al estilo de una cesta de picnic.

El Macintosh tiene un teclado tipo máquina de escribir, con un "tacto" excelente y adecuado para mecanografía al tacto. El teclado posee su propio procesador para tratamiento de funciones especiales y juegos de caracteres internacionales. El otro componente del "Mac", como se lo llama familiarmente, es el ratón. Llamado así en parte por el "rabo" que lo conecta con la unidad del sistema, este dispositivo manual, del tamaño de un paquete de cigarrillos, se desplaza por una superficie plana y nivelada. El cursor se desplaza en la pantalla de acuerdo al movimiento que el operador imprime al ratón sobre la superficie plana y se lo puede utilizar para seleccionar las actividades que el usuario desee que efectúe la máquina. Este enfoque al diseño de ordenadores se considera mucho más agradable para el usuario que el que emplean la mayoría de las máquinas, que exigen un conocimiento de instrucciones específicas de operatoria. Por ejemplo, si se deseara abrir un archivo de documentos, movería el ratón de modo que el cursor cayera sobre el pequeño símbolo gráfico (icono) que representa una hoja de papel. Pulsando entonces el botón del ratón la pantalla se abriría para esa actividad. Habiendo entrado su archivo desde el teclado, se utilizaría el ratón para retornar al menú principal de instrucciones de iconos y se podría guardar el archivo colocando el cursor sobre el símbolo que representa un disco.

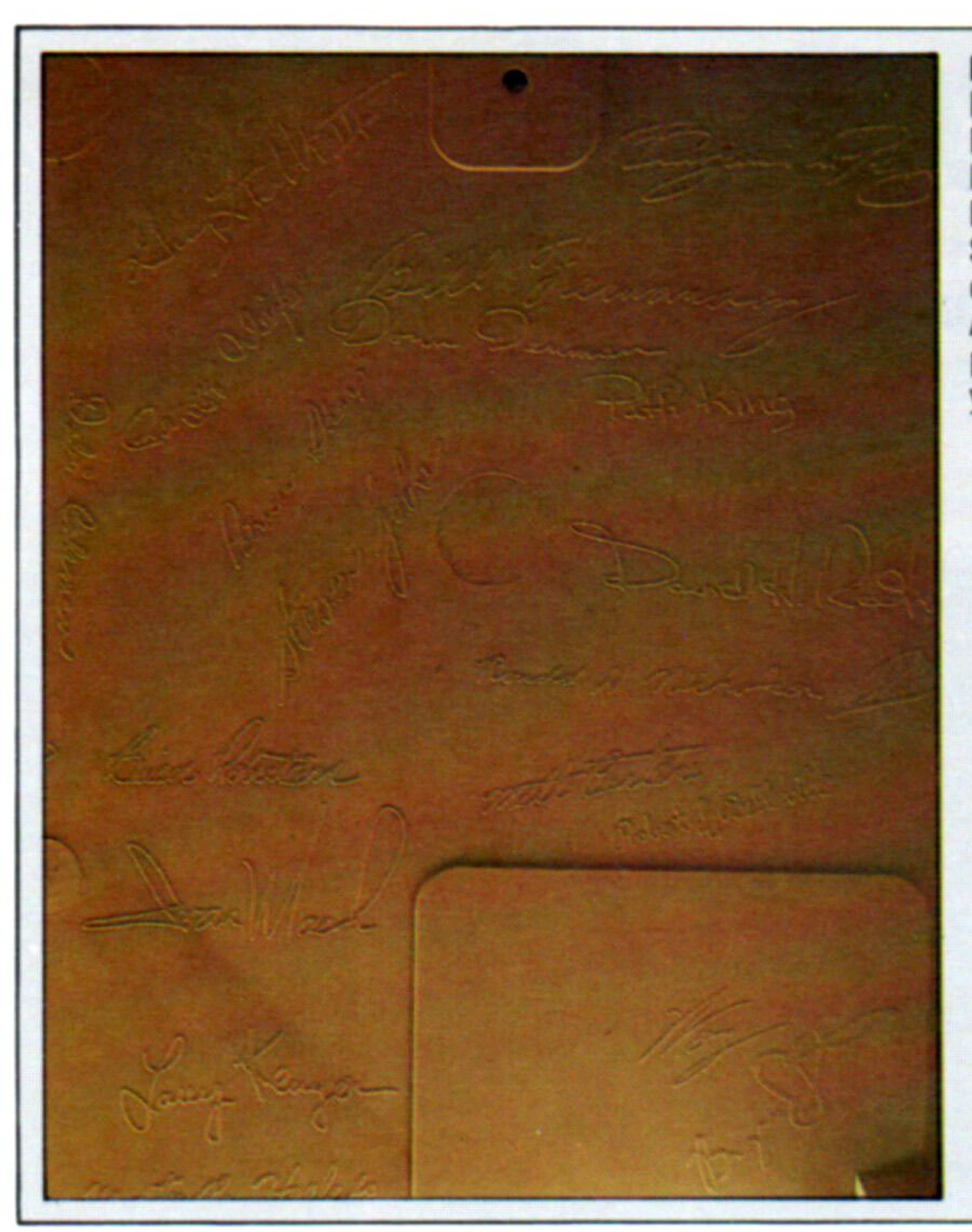
El Macintosh viene con 128 Kbytes de memoria para el usuario, que se pueden aumentar hasta 512 Kbytes mediante la sustitución de la RAM existente por chips de 256 Kbytes. El Mac también posee 64 Kbytes de ROM empaquetados ajustadamente con software operativo, que manipula virtualmente todas las operaciones del sistema, así como algunas características especiales. La unidad de disco Sony utiliza discos de 3 ½ pulgadas, que almacenan hasta 400 Kbytes por una cara y son más fiables que los discos de 5 ¼ pulgadas.

La pantalla del Macintosh es de 512 por 342 pixels y responde a un "mapa de bits", de modo que se puede acceder de forma individual a cada uno de sus más de 175 000 puntos. Ello hace posibles algunas aplicaciones de gráficos verdaderamente asom-



Ian McKinnell

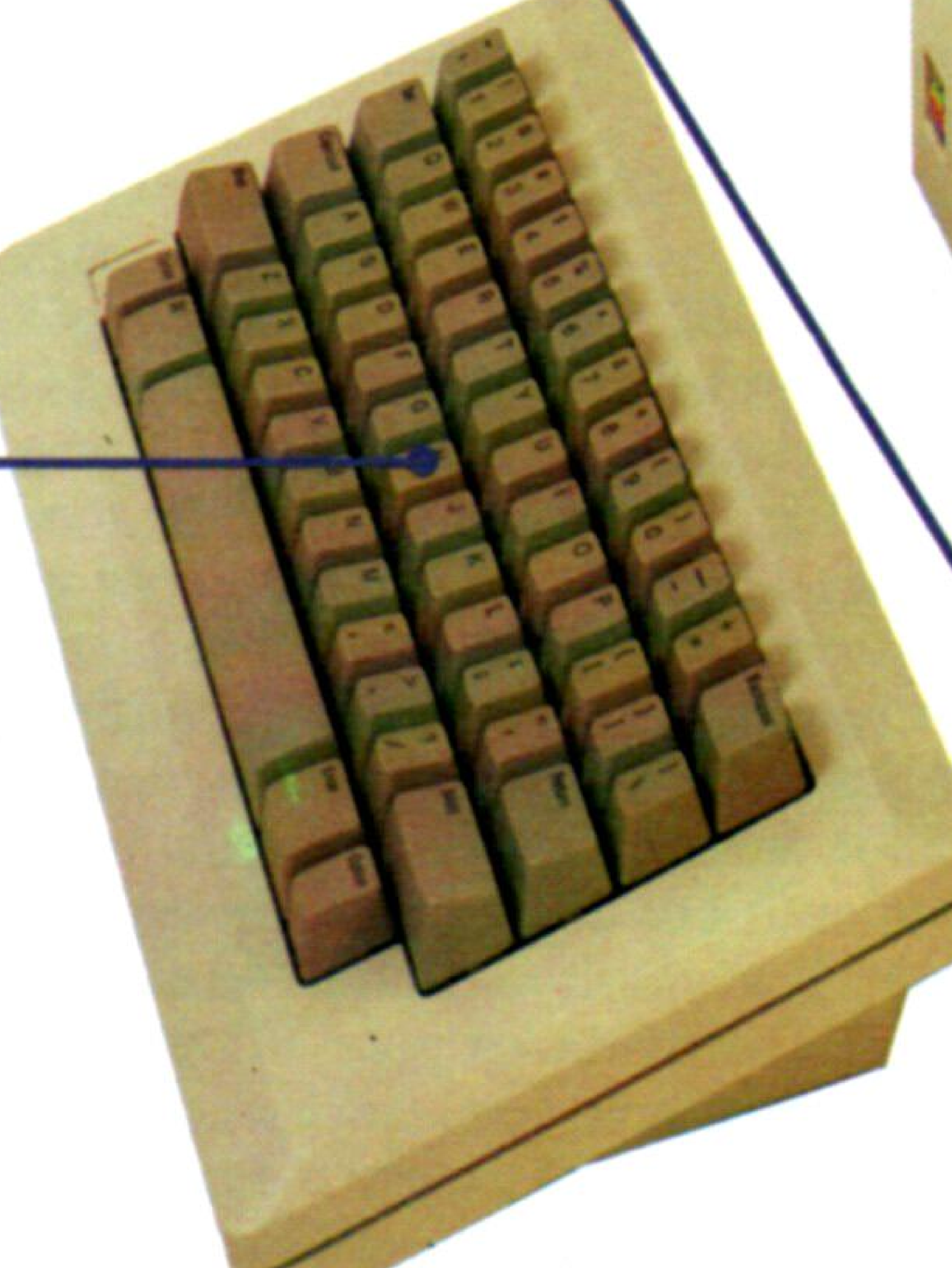
El sistema Macintosh
El Macintosh está diseñado para ocupar el menor espacio posible sobre un escritorio. La altísima resolución de la pantalla permite realizar gráficos que por lo general sólo son posibles en máquinas que cuestan 10 veces más que ésta



Panel de firmas
En el interior de la carcasa del Macintosh, Apple ha grabado las firmas del equipo de diseño. Entre ellas se incluyen las de Steven Jobs y Steve Wozniak, que fueron los creadores de Apple. Fue su genio lo que posibilitó la existencia del Lisa y el Macintosh

brosas. Además de proporcionar una enorme diversión, los trucos de gráficos del Macintosh tienen un gran valor para diseñadores, arquitectos, asesores, relaciones públicas, fotógrafos y muchos otros profesionales. Dado que el Mac está diseñado para funcionar específicamente con la impresora de Apple ImageWriter, de gran velocidad, todos sus espléndidos gráficos se imprimen exactamente igual a como aparecen en la pantalla.

No obstante la gran calidad y fiabilidad del hardware del Macintosh, es la solidez adicional de su software lo que hace que la máquina sea excepcional. Con la integración de hardware y software y las exhaustivas instrucciones operativas basadas en ROM, a las personas que desarrollan software les resulta relativamente sencillo transferir al Macintosh programas escritos para otros ordenadores. El ordenador es tan fácil de usar que literalmente se lo puede enchufar y ponerse en seguida a trabajar con él sin ningún conocimiento previo sobre operatoria de ordenadores.



Placa analógica
Esta placa controla la pantalla y la fuente de alimentación eléctrica. El Macintosh no tiene necesidad de ventilador. El exceso de calor se canaliza, a través de placas metálicas, hacia las ranuras de ventilación del mueble

Altavoz incorporado

Cabezal de la unidad de disco

Control de contraste de la pantalla

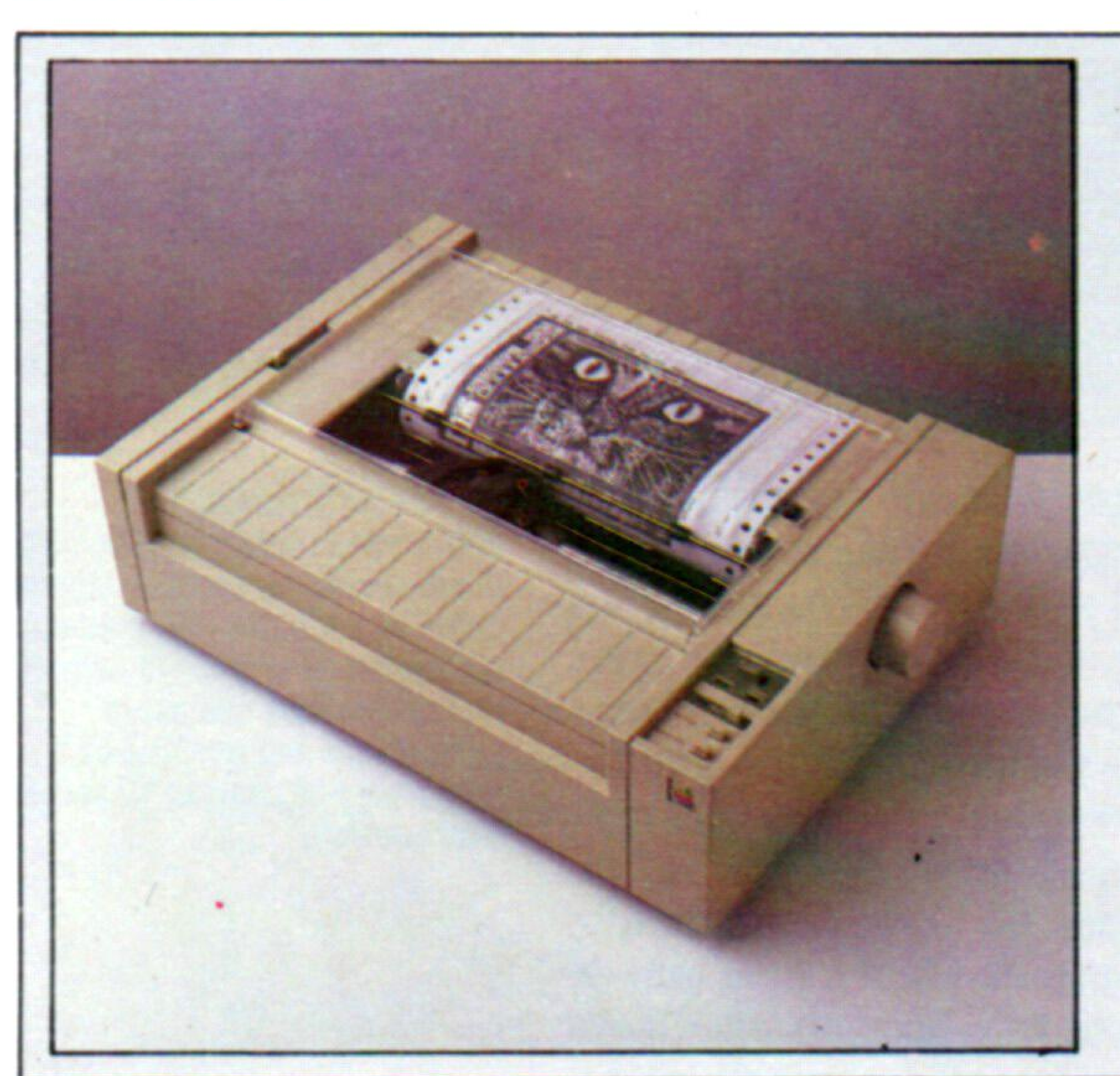
Unidad de disco Sony de 3 ½ pulgadas
Construida especialmente para Apple; esta unidad contiene 400 K en simple cara. Los discos de doble cara almacenarán, cuando estén disponibles, 800 K cada uno

RAM de video
Parte de los 22 K requeridos por la visualización de pantalla se toman de estos circuitos DMA (*direct memory access*)

Teclado
El teclado separado del Macintosh posee su propio procesador para tratar juegos de caracteres internacionales y funciones especiales. Debido a la existencia del ratón, no se necesitan teclas para el cursor

Conector del teclado

Ratón
Controla el movimiento del cursor y se utiliza para "seleccionar" objetos en la pantalla, actuando luego sobre ellos según instrucciones escogidas de menús



Mejorando la imagen

ImageWriter es una impresora en serie que genera texto a una velocidad de hasta 120 caracteres por segundo. Su velocidad es aún más evidente en la modalidad de gráficos, porque genera gráficos del tipo mapas de bits, siguiendo el formato de la pantalla



Conexión de analógico a digital
Un cable plano conecta las placas analógica y digital

Salida de audio

Salida para impresora

Bus en serie
El bus en serie, también llamado "ranura virtual", permite añadir diversos dispositivos periféricos

Conector para la unidad de disco externa

Conector del ratón

Adaptador E/S 6522
El 6522, también utilizado para el control de E/S en el Apple IIe, manipula el teclado, el ratón y los circuitos del reloj de "tiempo real"

Controlador de comunicaciones en serie

Controlador del disco
Este chip controla la unidad de disco Sony incorporada, así como la unidad externa, si la hay

CPU
Motorola 68000. Este chip realiza operaciones internas de 32 bits a la vez, pero envía y recibe datos de 16 bits

128 K de RAM para el usuario
Estos 16 chips se pueden reemplazar por chips de RAM de 256 K, que le proporcionarían al Macintosh un total de 512 K de memoria para el usuario. No obstante, 128 K son suficientes para las aplicaciones existentes

APPLE MACINTOSH

DIMENSIONES
343 x 254 x 254 mm
(mueble pantalla/unidad disco)

CPU
Motorola 68000, 7,83 MHz

MEMORIA
128 K de RAM, 64 K de ROM

PANTALLA
Monocromática, 512 x 342 pixels, ventanas, menús de selección, "iconos"

INTERFACES
Ratón, impresora, unidad de disco externa, amplificador hi-fi, bus en serie

LENGUAJES DISPONIBLES
BASIC, COBOL, PASCAL

TECLADO
Tipo máquina de escribir, 59 teclas, teclado numérico opcional

DOCUMENTACION
Existe un manual de operatoria con una cassette de audio y un disco de "enseñanza guiada". Se dan manuales para MacPaint y MacWrite, que incluyen enseñanza guiada mediante combinación de disco y cassette

VENTAJAS
El Mac posee un software muy potente y fácil de utilizar. El ratón hace que la operatoria resulte muy sencilla y directa. Los componentes, muy accesibles y estándares, hacen que el mantenimiento y la actualización resulten simples

DESVENTAJAS
El Mac es un ordenador pequeño muy caro y fuera del alcance de muchos usuarios. Con una sola unidad de disco, las operaciones sobre archivos son lentas y engorrosas. Existe carencia de software

Ayuda exterior

La segunda unidad de disco, externa, que Sony ha hecho a la medida para Apple, es un valioso accesorio para el Macintosh. Sin ella, muchas operaciones se retrasan debido a la necesidad de cambiar los discos. Esta unidad externa no requiere interface



Ian McKinnell

Apple realizó fuertes inversiones en la tecnología que permitió crear el Lisa y el Macintosh y está acercando esa tecnología al gran público con agresividad y orgullo. A la vista de su diseño y su construcción, el Macintosh es a todas luces un precursor del futuro.



Teclado
Los primeros envíos de máquinas Macintosh incluían el teclado norteamericano. El teclado de este ordenador posee un juego de caracteres internacionales completo incorporado, de modo que cualquier versión de Macintosh puede generar todos los caracteres especiales que exigen el alemán, francés, castellano, griego y otros idiomas

"Ventana escritorio"
La pantalla del Macintosh continúa la analogía del "escritorio" creada para el ordenador Lisa, y utiliza "ventanas" para mostrar porciones de documentos que son demasiado extensos como para caber en la pantalla. Las ventanas se pueden superponer unas sobre otras, como vemos aquí, de modo que utilizar el Macintosh es bastante parecido a trabajar sobre un escritorio

Ian McKinnell

Aguas turbulentas

Cocodrilos, anacondas, troncos flotantes y helicópteros que arrojan minas son algunos de los peligros que se deben sortear en este juego

River rescue (Rescate en el río) es un juego recreativo del más puro estilo de "disparos" y sin ninguna pretensión de ser nada más que eso. Lo produce Creative Sparks, la división de software de Thorn EMI, y el apoyo de una empresa de tal calibre resulta evidente. Se vende en versiones para cuatro ordenadores personales: Spectrum de 48 K, Commodore 64, Atari y Vic-20 sin ampliar, y se suministra en un paquete burbuja diseñado especialmente en vez de en la caja habitual de cassette de música. Con cada versión se incluye un pequeño pero muy útil folleto de instrucciones.

El juego en sí es básicamente muy simple, pero incorpora acción suficiente como para satisfacer a cualquier adicto a los juegos recreativos. El jugador controla la motora de rescate y tiene la misión de salvar a un grupo de científicos que se han quedado embarrancados en un tramo del río. En primer lugar, no se explican las causas por las cuales es necesario rescatar a los científicos, pero las instrucciones indican que deben ser trasladados a un hospital, de modo que presumiblemente han sido víctimas de algún tipo de accidente.

Mientras intenta recoger a los científicos heridos, el jugador debe conducir su barca, que navega a considerable velocidad, evitando islas y troncos flotantes, apartándose eventualmente de los cocodrilos que puedan aparecer. La versión para el Vic-20 es algo diferente, con peligros adicionales en forma de anacondas y canoas. A intervalos se ven a lo largo de la ribera varios muelles y en ellos se encontrarán los individuos a rescatar. La transferencia con éxito de un científico hasta el otro lado del río incrementa de forma considerable el marcador, pero también se pueden conseguir puntos matando a los cocodrilos que pululan por el río.

Se pueden conseguir puntos extras transfiriendo a los científicos por grupos, si bien la capacidad máxima de la barca es de nueve científicos. Ello hace que las cosas sean un poco más dificultosas, porque toda la tripulación se perderá si el barco choca contra algún obstáculo. Por consiguiente, hay que elegir entre un marcador elevado con el riesgo de perder todo o jugar seguro transfiriendo a los pasajeros de uno en uno. Para empeorar aún más las cosas, es probable que en cualquier momento aparezca un helicóptero (un avión, en la versión para el Spectrum) y arroje minas al agua, que se deben hacer estallar antes de poder seguir adelante.

La versión para el Vic-20 se suministra en formato de cartucho para evitar el tedioso proceso de cargar el cassette. Aquí se ofrece la opción de tres o seis científicos embarrancados y se dispone de seis vidas por juego. Además, los puntos conseguidos en cada "vida" se pasan a las subsiguientes "reencarnaciones", lo que simplifica considerablemente las cosas. En esta versión, no obstante, se tienen tres ríos extras por los cuales se puede navegar.

River rescue es un juego de acción pura y del tipo en el que hay que disparar contra todo lo que se mueva, y se ha diseñado con cuidado para que resulte suficientemente difícil de jugar como para mantenerlo a uno ocupado durante un tiempo, si bien se puede argüir que carece de la imaginación necesaria para hacer del mismo algo realmente especial.

River rescue: Para el Spectrum de 48 K, el Atari, el Commodore 64 y el Vic-20 (sin ampliar)
Editado por: Creative Sparks, 1st Fllor, Thomson House, 296 Farnborough Road, Farnborough, Hants., Gran Bretaña
Autor: Kevin Buckner
Palancas de mando: Kempston/Interface 1 (Spectrum)
Compatibles con Commodore (Vic-20 y Commodore 64)
Compatibles con Atari
Formato: Cassette;
cartucho (sólo el Vic-20)



¡Al rescate!
En la ilustración vemos *River rescue* en un Spectrum. La primera pantalla muestra la página de título, que proporciona buenos indicios sobre los gráficos que aparecerán. En la segunda pantalla vemos el juego ya ejecutándose. El barco transporta a un científico hacia lugar seguro, donde se unirá a los otros que han sido rescatados

Liz Heaney

Salir del círculo

En esta ocasión analizaremos la "recursión", una técnica que se utiliza en programación avanzada

La mejor manera de resumir el tema de esta investigación es mediante una definición que se suele dar comúnmente en broma:

Recursión: *véase* Recursión

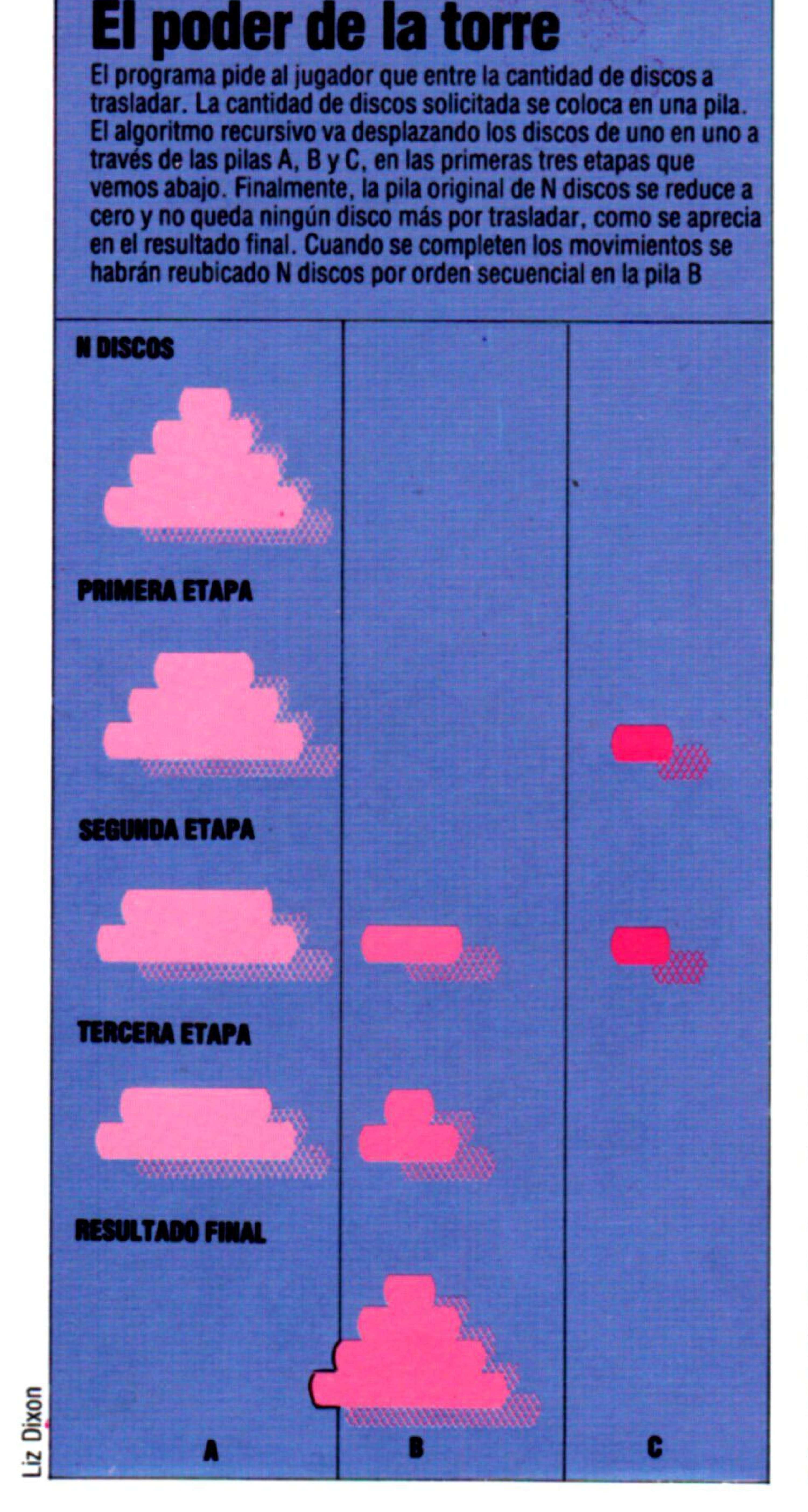
Esta definición circular demuestra un aspecto esencial de la recursión: algo que se define en sí mismo. Pero omite otro importante aspecto: para que la recursión sea operativa, debe haber una forma de salir de la circularidad.

El puzzle que hemos utilizado para ilustrar la recursión es *Las torres de Hanoi*. Comienza con un número de discos apilados por orden de tamaño, con el disco más grande en la parte inferior de la pila y el disco más pequeño arriba. Para resolver el puzzle se deben trasladar todos los discos de una pila a otra de acuerdo a las siguientes reglas:

1) Sólo se puede trasladar un disco cada vez;
2) No se puede colocar un disco sobre otro más pequeño;
3) No puede haber nunca más de tres pilas de discos.

El diagrama ilustra cómo utilizamos el concepto de recursión para que el problema sea manejable. Empezamos con una pila de cuatro discos. Asignándole a una variable N el valor de cuatro, indicamos el número total de discos que se deben trasladar. Dado que las reglas prohíben mover más de un disco por vez, utilizamos una fórmula recursiva para reducir el valor de N en uno, continuando luego el cálculo hasta que N sea igual a 1. Cuando N = 1, el programa interrumpe el cálculo y traslada el disco apropiado.

Trabajando con una versión de BASIC que permita la recursión es fácil escribir un programa que siga exactamente el proceso que hemos descrito. En el programa en BASIC BBC, todo el trabajo de calcular los movimientos se realiza entre las líneas 1000 y 1050. ¡El resto del programa sólo produce la visualización gráfica en movimiento!



La versión para el Spectrum

Para convertir el programa *Las torres de Hanoi* al BASIC del Spectrum tenemos que reemplazar un procedimiento recursivo por una subrutina recursiva, que comienza en la línea 1000 de nuestro listado. Cada vez que la subrutina tiene que hacer una llamada recursiva a las matrices M, A, B o C incrementa la variable puntero J y coloca los nuevos valores en M(J), A(J), B(J) y C(J). Posteriormente, estos nuevos valores se pueden utilizar en la siguiente llamada a la subrutina sin alterar los valores antiguos. Al final de la subrutina, se decrementa el valor de J, restaurando, por consiguiente, los valores antiguos. Este método siempre se puede emplear para escribir subrutinas recursivas en BASIC, independientemente de lo complicada que sea la recursión.

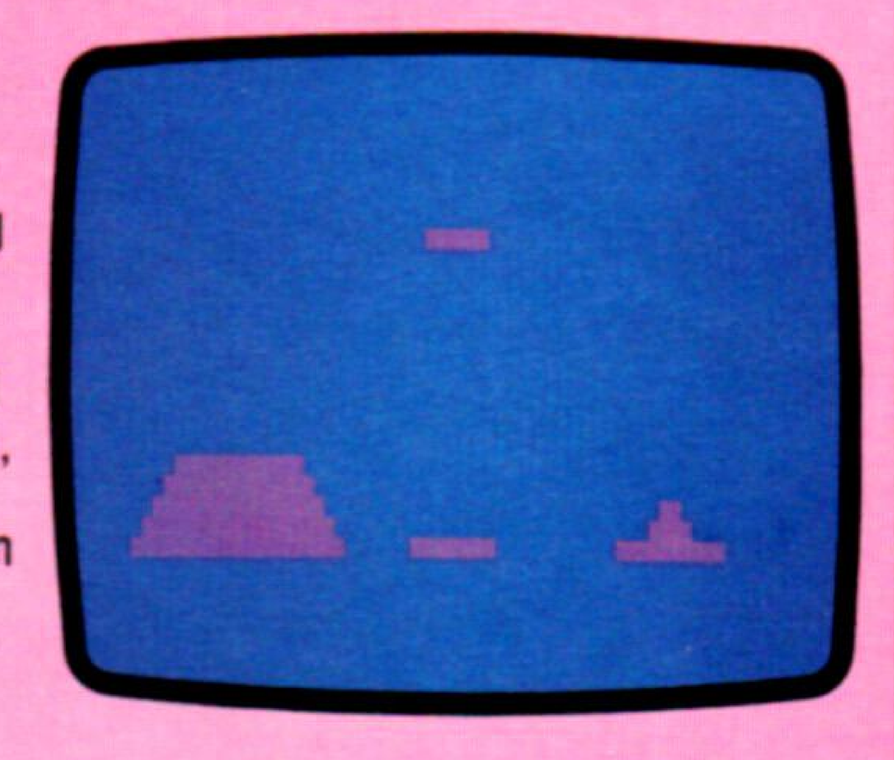
La sección del programa para la visualización es directa, imprimiendo un objeto en una nueva posición y borrándolo mediante la impresión de caracteres negros en la posición antigua. El programa muestra la sección transversal de una pila de discos. Para que las pilas parezcan simétricas, hemos terminado cada barra con caracteres de gráficos hechos mitad con un espacio y mitad con un color sólido.

BBC Micro

```
10 DIM M(10): DIM A(10): DIM B(10): DIM C(10)
20 DIM D$(10,10): DIM H(3): DIM P(3,10)
30 GO SUB 3000
90 DIM M(100): DIM A(100): DIM B(100): DIM C(100)
100 INPUT "CUANTOS DISCOS? ";N
110 IF N < 1 OR N > 10 THEN GO TO 100
120 GO SUB 3100
130 LET J = 1: LET M(J) = N: LET A(J) = 1: LET
    B(J) = 2: LET C(J) = 3
140 GO SUB 1000
200 STOP
1000 IF M(J) = 1 THEN GO SUB 1500: RETURN
1010 LET J = J + 1
1020 LET M(J) = M(J-1)-1
1030 LET A(J) = A(J-1)
1040 LET B(J) = C(J-1)
1050 LET C(J) = B(J-1)
1060 GO SUB 1000
1100 LET M(J) = 1
1110 LET A(J) = A(J-1)
1120 LET B(J) = B(J-1)
1130 LET C(J) = C(J-1)
1140 GO SUB 1000
1200 LET M(J) = M(J-1)-1
1210 LET A(J) = C(J-1)
1220 LET B(J) = B(J-1)
1230 LET C(J) = A(J-1)
1240 GO SUB 1000
1300 LET J = J-1
1310 RETURN
1500 LET PA = A(J): LET PB = B(J)
1510 LET M$ = D$(P(PA,N + 1-H(PA)))
1520 FOR I = 22 - H(PA) TO 7 STEP -1
1530 PRINT AT I -1,10*(PA - 1.;M$;
1540 PRINT AT I,10*(PA-1); B$;
1550 NEXT I
1560 FOR I = 10 * (PA -1) TO 10*(PB - 1) STEP SGN (PB - PA)
1570 PRINT AT 6,I;M$;
1575 PRINT AT 6,I;B$;
1580 NEXT I
1590 FOR I = 6 TO 20 - H(PB)
1600 PRINT AT I,10*(PB - 1);B$;
1610 PRINT AT I + 1,10*(PB - 1);M$;
1620 NEXT I
1640 LET H(PB) = H(PB) + 1: LET P(PB,N +
     1-H(PB)) = P(PA,N + 1 - H(PA))
1650 LET P(PA,N + 1 - H(PA)) = 0: LET H(PA) = H(PA)-1
1660 RETURN
3000 LET B$ = "                    ": LET C$ = CHR$
     143 + CHR$ 143 + CHR$ 143 + CHR$143
3010 LET C$ = "": FOR I = 1 TO 10: LET
     C$ = C$ + CHR$ 143: NEXT I
3020 FOR I = 1 TO 9 STEP 2
3030 LET D$(I) = B$( TO 4 - INT (I/2)) +
     CHR$ 133 + C$ ( TO 2 * INT (I/2)) + CHR$
     138 + B$( TO 4-INT (I/2))
3040 LET D$ (I + 1) = B$( TO 4 - INT (I/2)) +
     C$( TO I + 1) + B$( TO 4-INT (I/2))
3050 NEXT I
3060 RETURN
3100 INK 3: PAPER 6: BORDER 6: CLS
3120 FOR I = 1 TO N
3130 PRINT AT 21-N + I,0;D$(I);
3135 LET P(1,I) = I; LET P(2,I) = 0: LET P(3,I) = 0
3140 NEXT I
3150 LET H(1) = N: LET H(2) = 0: LET H(3) = 0
3160 RETURN
```

Spectrum

```
10 DIM D$(12),H(3),P(3,12)
20 PROCINIC
100 INPUT "CUANTOS DISCOS?(1-12)";N
110 IF N < 1 OR N > 12 THEN 100
120 PROCVISUALIZ(N)
130 PROCHANOI(N,1,2,3)
200 END
1000 DEFPROCHANOI(M,PA,PB,PC)
1010 IF M = 1 THEN PROCMOVER(PA,PB):
     ENPROC
1020 PROCHANOI(M-1,PA,PC,PB)
1030 PROCHANOI(1,PA,PB,PC)
1040 PROCHANOI(M-1,PC,PB,PA)
1050 ENDPROC
1100 DEFPROCMOVER(PA,PB)
1110 D$ = D$(P(PA,N + 1 - H(PA)))
1120 FOR I = 24-H(PA) TO 10 STEP -1
1130 PRINT TAB(13*(PA-1),I);B$;
1140 PRINT TAB(13*(PA-1),I-1);D$;
1150 NEXT I
1160 FOR I = 13*(PA-1) TO 13*(PB-1) STEP
     SGN(PB-PA)
1170 PRINT TAB(I,9);D$
1180 NEXT I
1190 FOR I = 9 TO 22-H(PB)
1200 PRINT TAB(13*(PB-1),I);B$
1210 PRINT TAB(13*(PB-1),I + 1);D$;
1220 NEXT I
1240 H(PB) = H(PB) + 1:P(PB,N + 1-H(PB)) =
     P(PA,N + 1-H(PA))
1250 P(PA,N + 1-H(PA)) = 0: H(PA) = H(PA)-1
1260 ENDPROC
3000 DEFPROCINIC
3020 FOR I% = 1 TO 11 STEP 2
3030 D$(I%) = CHR$150 + STRING$(5-I%DIV2," ")
     + CHR$234 + STRING$(2*(I%DIV2),CHR$255)
     + CHR$53 + STRING$(5-I%DIV2," ")
3040 D$(I% + 1) = CHR$150 + STRING$(5 -
     I%DIV2," ") + STRING$(I% + 1,CHR$255) +
     STRING$(5-I%DIV2," ")
3050 NEXT I%
3060 B$ = CHR$150 + STRING$(12," ")
3070 VDU 23,1,0;0;0;0;
3080 ENDPROC
3100 DEFPROCVISUALIZ(N)
3110 CLS
3120 FOR I = 1 TO N
3130 PRINT TAB (0,23-N + I);D$(I);
3135 P(1,I) = I:P(2,I) = 0: P(3,I) = 0
3140 NEXT I
3150 H(1) = N:H(2) = 0:H(3) = 0
3160 ENDPROC
```



Problemas recursivos
Ésta es una fotografía del programa *Las torres de Hanoi* ejecutándose en el Spectrum. El color de los bloques se puede cambiar muy fácilmente. Si intenta seguir el modo en que el ordenador resuelve el problema, observe con gran atención, ¡porque las acciones se suceden con bastante rapidez!

Ian McKinnell

El diseño clave

Aquí estudiaremos cómo utilizar los módulos en el desarrollo de un programa completo

Cuando se construye un programa, es una buena idea desarrollar una estructura global, compuesta de un nivel básico de rutinas de uso general que son empleadas por otras rutinas de creciente especialización a niveles superiores, todo bajo la dirección de un único módulo de control ubicado en la parte superior. Esta estructura "piramidal" nos permitirá utilizar un método de diseño denominado *refinamiento de programas* o *diseño top-down* (de arriba abajo).

El diseño top-down, como su nombre indica, implica diseñar primero el programa de control de más alto nivel. Sus funciones se describen en términos de llamadas a rutinas de nivel "inferior" y, por el momento, no es necesario que nos preocupemos demasiado sobre cómo funcionarán estos módulos de nivel inferior. Después de hacer esto, nos desplazamos un nivel hacia abajo y describimos el funcionamiento de cada una de las rutinas a las que llama el módulo de mayor nivel. Cada rutina se describe en términos de las rutinas que debe llamar, y este proceso se repite nivel a nivel hasta que llegamos al nivel inferior.

En esta etapa, las funciones que realiza la rutina que estamos describiendo son tan simples que se pueden definir utilizando el propio lenguaje de programación.

Como ejemplo, analicemos el diseño del juego *El ahorcado*. En vez de que el jugador trate de adivinar una palabra escogida por el programa, como sucede en la mayoría de las versiones para ordenador que existen de este juego, deseamos que el programa adivine una palabra que hemos elegido nosotros. Una forma de conseguirlo, sin proporcionar al programa una extensa lista de palabras en castellano, consiste en entrar los datos relativos a las probabilidades de aparición de secuencias determinadas de letras.

```
100 REM Inicializar variables y matrices


500 REM ***** Rutina de control **********
510 REM
520 GOSUB 1000: REM Pantallas de títulos y ayudas
530 GOSUB 2000: REM Preparar tablero
540 GOSUB 4000: REM Obtener del jugador la longitud de la
    palabra
550 GOSUB 8000: REM Seleccionar conjunto de datos y cargarlo
560 GOSUB 3000: REM Adivinar una letra
570 GOSUB 4500: REM Verificar conjetura con el jugador
580 GOSUB 5000: REM Actualizar el tablero
590 IF JUEGO—NO—TERMINADO THEN 560: REM Hacer
    nuevas conjeturas hasta que termine el juego
600 IF GANADO THEN GOSUB 10000 ELSE GOSUB 11000: REM
    Dar final adecuado para ganado o perdido
610 GOSUB 6000: REM ofrecer al jugador otra partida
620 IF OTRA THEN 530: REM si otra entonces volver a empezar
630 GOSUB 7000: REM decir adios y parar
640 END
```

Antes de empezar sabemos que se deben hacer varias cosas: es necesario inicializar las variables, se han de dimensionar las matrices, se debe preparar la visualización del "tablero" y actualizarla cuando sea necesario, y se han de escribir rutinas que lleven el marcador, hagan conjeturas y que den por terminado el juego.

Nuestro primer intento por diseñar la rutina de control posee una simple sentencia REM para indicar que se deben inicializar variables y matrices; podemos ir colocando todos los detalles en una etapa posterior. La rutina de control en sí misma es sencillamente un par de bucles. El bucle exterior (línea 620) verifica si el usuario está indicando el fin de una sesión, mientras que el bucle interior (línea 590) verifica si se ha acabado el juego.

Si necesitáramos comprobar la rutina de control, deberíamos preparar subrutinas ficticias para las GOSUB. Cada GOSUB de la rutina de control debe tener una sentencia REM que explique su función y ha de comenzar en un número de línea conveniente, preferiblemente uno que sea una cifra redonda, como 1000 o 5000. Es una buena idea asegurar que todas las rutinas de funciones similares tengan números de línea estandarizados; esto hará que las cosas sean más fáciles cuando se trasladen rutinas de un programa a otro. Por ejemplo, las instrucciones del juego podrían incluirse en una subrutina que empiece en la línea 1000, mientras que una línea de programa GOSUB 7000 puede dar siempre un juego por terminado llamando a una rutina estándar.

Nuestra rutina de control inicial es corta y sencilla. Cabrá en la pantalla y, por lo tanto, será más fácil de comprender y depurar que un programa que ocupe varias pantallas. Las tres variables, JUEGO-NO-TERMINADO, GANADO y OTRA, son todas indicadores o flags que se establecen en las diversas rutinas a las que llama la rutina de control y se utilizan aquí para determinar si el programa de control funciona del modo que nosotros pretendemos. En esta simple rutina de control sería bastante fácil detectar cualquier posible error de lógica.

En esta etapa es preciso analizar la estructura del programa con ojo crítico: necesitamos asegurarnos de que éste se comporte como debe en todas las circunstancias. También podemos comenzar a introducir mejoras en el diseño del programa; por ejemplo, tal vez queramos poder disponer de las instrucciones en cualquier etapa del juego, y también sería una buena idea llevar un registro de cuántas partidas han ganado el ordenador o el jugador y una lista de las palabras que vencen al programa. Cualquiera de estas modificaciones, o todas ellas, se pueden efectuar en esta etapa.

El siguiente paso consiste en especificar cada una de las subrutinas. Nuestros listados muestran el as-



pecto que podrían tener dos de estas subrutinas. La primera (que comienza en la línea 4000) simplemente solicita al usuario un número entre 1 y 20 (la longitud de la palabra). Utiliza una subrutina de uso general que da por sentado existe en la línea 51000, que tomará una serie especificada en PROMPT$, la imprimirá y luego aceptará una entrada numérica hecha por el usuario. Si este número no es un entero comprendido entre los límites establecidos por MIN% y MAX%, generará un mensaje de error y pedirá al usuario que entre un nuevo número. Esta subrutina se podría utilizar fácilmente en otros programas y se podría crear una biblioteca de este tipo de módulos de usos generales para emplearlos en proyectos ulteriores.

```
4000 REM Descubrir longitud de la palabra del jugador
4010 REM
4020 PROMPT$ = "Cuantas letras tiene su palabra?"
4030 MIN% = 1
4040 MAX% = 20
4050 GOSUB 51000: REM entrar un numero entero entre MIN%
     y MAX%
4060 LONGPAL% = RESP%: REM RESP% es usada por la
     subrutina de la 51000 para devolver la respuesta
4070 RETURN
```

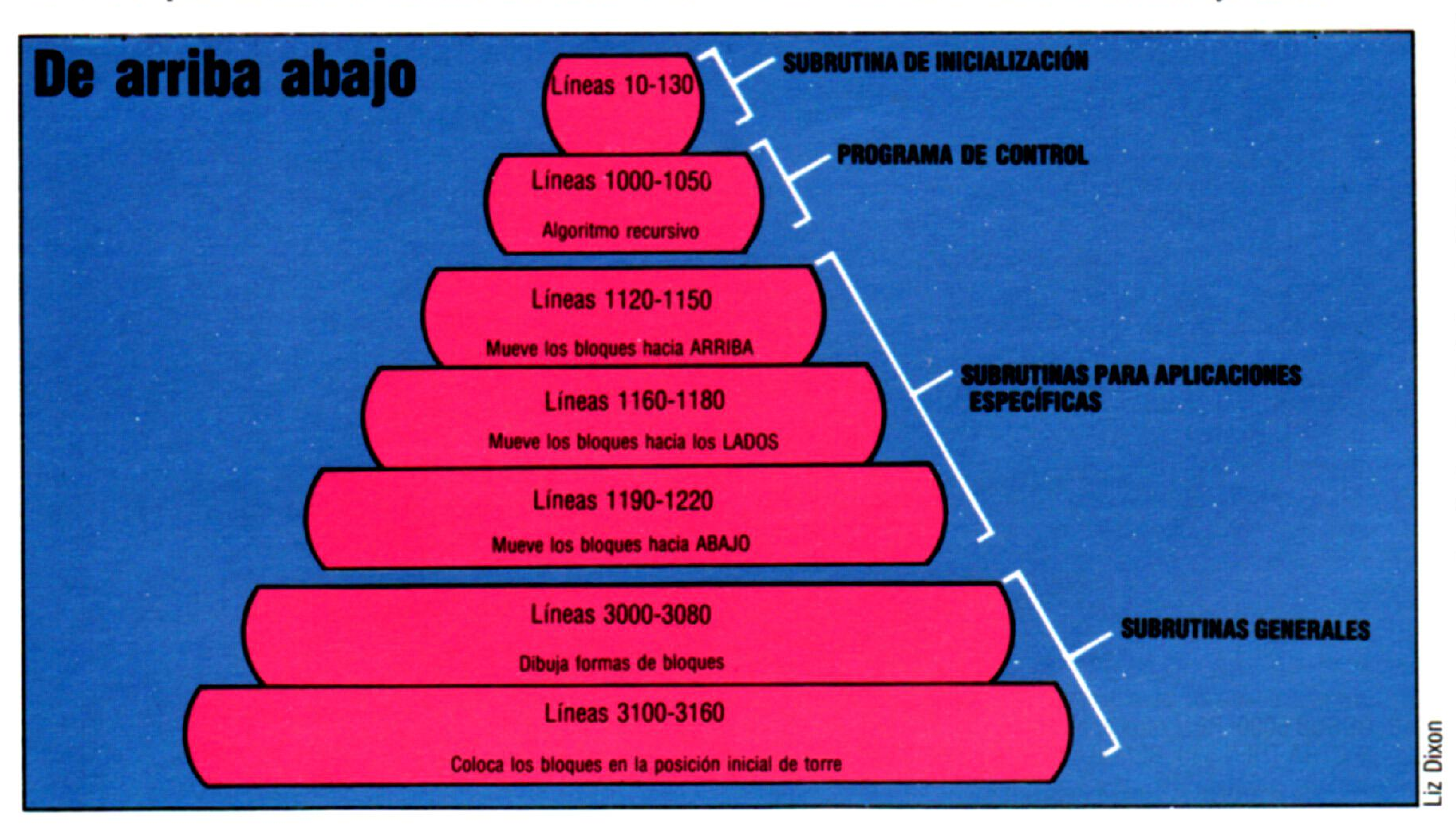
```
8000 REM seleccionar juego datos y cargarlo
8010 REM
8020 IF LONGPAL% > 7 THEN FICH—L% = 8
                     ELSE FICH—L% = LONGPAL%
8030 FICHNO—L$ = STR$(FICH—L%)
8040 NOMFICH$ = "TABLA" + FICHNO—L$
8050 GOSUB 9000:REM OPEN,READ & CLOSE el fichero con
     datos de probabilidad para la longitud de palabra
     adecuada.
8060 RETURN
```

La otra rutina (que empieza en la línea 8000) utiliza variables locales (FICH–L% y FICHNO–L$). Hemos dado por sentado que los datos necesarios para adivinar una letra están en ocho juegos de tablas que dan las probabilidades de hallar una letra determinada junto a cualquier otra. Dado que sólo deseamos tener un juego de datos en RAM en cualquier momento dado, debemos construir una serie en NOMFICH$ para retener el nombre del archivo de datos y después llamar a la subrutina de la línea 9000 para que lea el archivo.

En muchos casos nos encontraremos con que nuestro programa pasa directamente de una rutina a otra. Sin embargo, por lo general será mejor crear una rutina extra que llame a las otras dos sucesivamente. Esto puede parecer una complicación innecesaria, pero nos permite mantener un firme control sobre el "flujo" del programa y ofrece la ventaja adicional de mantener separados los módulos del programa de modo que se los pueda agregar fácilmente a otros programas.

Este uso de las subrutinas que son transportables de un programa a otro implica un trabajo extra y se debe tener cuidado al diseñar las rutinas, de modo que sean aptas para utilizarse en una amplia gama de circunstancias. Con frecuencia esto se puede conseguir simplemente reemplazando constantes por variables. Es importante que todas las subrutinas estén bien documentadas. La documentación ha de especificar el objetivo exacto de la rutina, dando detalles de las variables empleadas, los valores que se esperan como entrada y salida y cualquier posible efecto colateral (desplazar la posición del cursor, cambiar el mapa de memoria, cerrar archivos, etc.).

Un trazado estándar también resulta muy útil; debe asegurarse de que todos los números de línea tengan un intervalo fijado, que los títulos y comentarios se limiten a un grupo de números de línea al comienzo de la rutina y que el RETURN esté siempre en la última línea. Asegúrese de apuntar el primer y último número de línea de cada rutina. Cuando se requiere una rutina de la biblioteca, verifique que el programa tenga un agujero apropiado entre sus números de línea y luego mezcle (MERGE) la subrutina con el programa. Si su micro no posee instrucción MERGE, quizá sea posible utilizar un editor de textos para combinar programas que hayan sido guardados (SAVE) en formato ASCII en vez de en la forma "distintivada" habitual. Si esto no fuera posible, sería necesario digitar las subrutinas de su biblioteca cada vez que las necesite. No obstante, el solo hecho de que no sea necesario volver a diseñarlas bien vale el esfuerzo del trabajo extra.



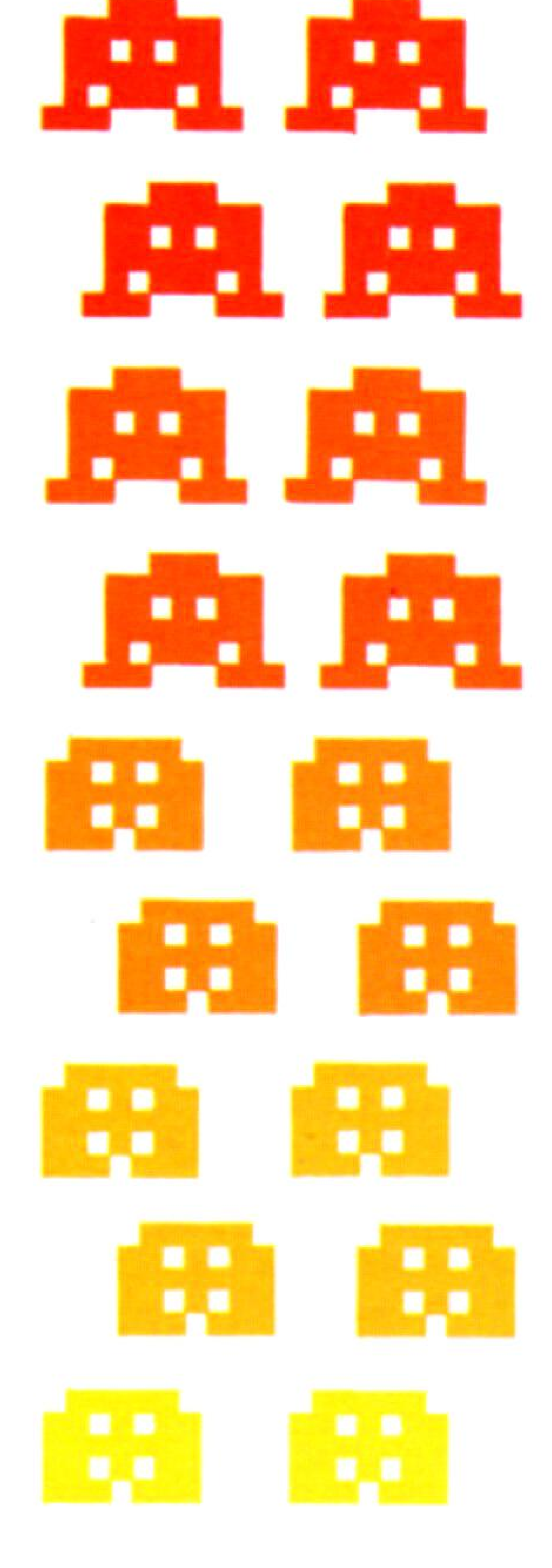
Programación top-down (de arriba abajo)
Este programa ilustra el principio de la programación top-down. Hemos utilizado el programa *Las torres de Hanoi*, que presentamos en la página 955. Los números de línea del diagrama corresponden al listado para el BBC.
El primer estrato de la estructura representa el programa de inicialización, que debe completarse antes de que se pueda ejecutar el resto del programa. El PROGRAMA DE CONTROL de nuestro diagrama representa el algoritmo recursivo, que realiza los cálculos y llama a las otras subrutinas cuando ello es necesario. Las SUBRUTINAS PARA APLICACIONES ESPECIFICAS (de la línea 1120 a la 1220) se utilizan para trasladar las formas de bloques de pila a pila en la visualización. Las dos secciones finales del diagrama, SUBRUTINAS GENERALES, representan las dos últimas secciones del programa que se emplean para formatear la visualización inicial y crear el diseño de los bloques. Compare esta estructura con el listado y verá que el programa está construido exactamente siguiendo esta secuencia

No tan rápido

A veces es necesario que el programa no se ejecute tan aprisa. He aquí uno de los métodos más populares para retardarlo

En el assembly del 6502 se pueden introducir bucles de retardo de diversas formas. El método más simple y natural consiste en cargar uno de los registros índice con un valor y decrementarlo progresivamente hasta que llegue a cero:

BUCLE RETARDO	TIEMPO EMPLEADO EN CADA OPERACIÓN
LDY #$07	Dos ciclos
DEY	Dos ciclos
BNE LOOP	Dos ciclos (3 ciclos si se bifurca a la misma página; 4 ciclos si la página es distinta)



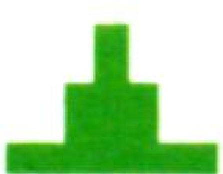
Cada instrucción en lenguaje máquina requiere un determinado número de ciclos de reloj para ser ejecutada. A veces en las descripciones de cómo operan las instrucciones es posible encontrar información sobre tales ciclos. Por ejemplo, la instrucción DEY necesita dos ciclos y LDY también cuando se emplea el modo de direccionamiento inmediato. Dado que cada ciclo tarda un microsegundo (millonésima de segundo), podemos calcular el "tiempo real" empleado en ejecutar el bucle de retardo. El número total de ciclos se calcula así:



1) La instrucción LDY #$07 tarda dos ciclos.
2) El programa da siete vueltas. Por cada vuelta la operación BNE emplea tres ciclos: luego las instrucciones DEY y BNE emplean (2 + 3) × 7 = 35 ciclos.
3) Puesto que el último BNE no provoca la vuelta, sólo emplea dos ciclos.

Número total de ciclos = 2 + 35 − 1 = 36. Así pues, el tiempo que consume este retardo es de 36 microsegundos.

Hay varios problemas asociados con el uso de bucles de retardo en código máquina para obtener retrasos "de tiempo real" (o sea, aquellos que pueden ser medidos en segundos o microsegundos con toda precisión). El primero y más importante es que mientras un procesador está ejecutando un programa en código máquina regularmente suspende esta actividad para efectuar servicios requeridos por otras partes del sistema, tales como inspeccionar el teclado, actualizar el reloj interno, etc. Estas discontinuidades en la ejecución del programa son conocidas como "interrupciones". En el chip del 6502 se producen dos tipos de interrupciones: NMI (*Non-Maskable Interrupt*: interrupción no enmascarable) e IRQ (interrupción solicitada). El nombre dado al primer tipo de interrupción implica que nada se puede hacer para evitar (enmascarar) tales interrupciones. Sin embargo, es posible detener las interrupciones IRQ, pues no son esenciales para el funcionamiento del procesador.



Las interrupciones IRQ pueden ser enmascaradas poniendo a 1 un determinado bit del registro indicador de estado. Esto se consigue con la instrucción SEI. Las interrupciones IRQ se vuelven a permitir devolviendo al registro el bit afectado a través de CLI. Enmascarando las interrupciones IRQ antes de entrar en el bucle de retardo, mejoramos su exactitud. Si ocurren interrupciones no enmascarables durante la ejecución del bucle, se producirán errores en el período de retardo. Nuestro bucle inicial presentará este otro listado que evita interrupciones IRQ:

INSTRUCCIÓN	FUNCIÓN	DURACIÓN
SEI	Desactiva IRQ	Dos ciclos
LDY = $07		36 ciclos
DEY		
BNE LOOP		
CLI	Restablece IRQ	Dos ciclos

La tarea de enmascarar las IRQ de esta forma añade cuatro ciclos a la rutina, con lo que su duración será de 40 microsegundos, siempre que no aparezcan interrupciones NMI.

Otro aspecto de los bucles de retardo es el de la "resolución", es decir, cómo varía el tiempo empleado en ejecutar el bucle entre un valor del contador y el siguiente. En nuestra rutina hemos cargado el registro Y con el valor siete, pero si hubiéramos escogido el valor seis, el retardo sería de 35 microsegundos (2 + 2 + (2 + 3) × 6 − 1 + 2). Si se escogiera el valor cinco duraría 30 microsegundos y así sucesivamente hasta un mínimo de resolución de 5 microsegundos.

Es posible "afinar" nuestro programa (para que el temporizado deje de ser múltiplo de cinco) colocando instrucciones NOP fuera del bucle. La instrucción NOP significa que el procesador no realizará operación alguna (N*o* OP*eration*), lo que hará en dos ciclos. Si deseáramos crear un retardo de 44 microsegundos, por ejemplo, basta con añadir dos instrucciones NOP a nuestro programa antes del bucle (o después):

SEI	Dos ciclos
LDY #$07	Dos ciclos
NOP	Dos ciclos
NOP	Dos ciclos
DEY BNE LOOP	34 ciclos
CLI	Dos ciclos



Este tipo de retardo tiene un tiempo límite determinado por el máximo valor que puede tomar Y. Dado que el registro índice Y es de ocho bits, el valor máximo es 255. Lo que nos da un tiempo máximo de 1 280 microsegundos (2 + 2 + (2 + 3) × 255 − 1 + 2), aproximadamente una milésima de segundo. A nuestros ojos es un tiempo brevísimo, pero no a los del procesador. A veces son necesarios retardos de mayor duración. Lograremos alargarla agregando instrucciones NOP *dentro* del bucle. Si, por ejemplo, incluimos una instrucción NOP en el bucle el retardo máximo pasa a ser de 1 790 microsegundos (2 + 2 + (2 + 2 + 3) × 255 − 1 + 2).

Para demoras bastante mayores que éstas deberemos diseñar algún otro método. Dos de los modos más frecuentes de producir largos retardos consisten en utilizar un segundo bucle anidado al primero, o bien producir decrementos en un valor numérico mucho mayor, por ejemplo uno de 16 bits logrado mediante dos bytes de la memoria. Cada uno de estos procedimientos tiene su período de resolución que es posible calcular.

Contador de bucle anidado

```
DELAY SEI
      LDX #$04     ;emplea registro X como contador bucle exterior
LOOP1 LDY #$FF     ;emplea registro Y como contador bucle interior
LOOP2 DEY          ;salida del bucle interior
      BNE LOOP2
      DEX
      BNE LOOP1    ;salida del bucle exterior
      CLI
```

El bucle interior del programa emplea 1 276 microsegundos (2 + (2 + 3) × 255 − 1) en su ejecución. El bucle exterior controla la ejecución del interior y realiza DEX y BNE cuatro veces. El tiempo total de este retardo es calculable: 2 + 2 + (1 276 + 2 + 3) × 4 − 1 + 2 = 5 129 microsegundos.

Retardos en el Z80

Cada instrucción en código máquina en el Z80 tarda un tiempo diferente en ser ejecutada (medido en unidades llamadas *estados T*). Además, el Z80 va a diferentes velocidades según las máquinas. Para calcular el tiempo real empleado por cada instrucción se divide el número de estados T de cada instrucción por la frecuencia de reloj del micro. Por ejemplo, una instrucción que emplee cuatro estados T en ser ejecutada por un procesador con 2 MHz de frecuencia de reloj tardará en ejecutarse dos microsegundos.

El libro de Rodnay Zak *Programming the Z80 (La programación del Z80)* contiene las duraciones de todas las instrucciones del Z80. Éstas son las velocidades de reloj de la CPU de las máquinas más populares que incorporan un Z80; el ZX81 (3,25 MHz), el Spectrum (3,5 MHz), el Tandy TRS80/Video Genie (1,7 MHz) y el Amstrad (4 MHz).

Podemos usar la instrucción NOP si deseamos un retardo de muy corta duración. En un micro de 2 MHz esta instrucción producirá un retardo de dos microsegundos. Podemos usarlas sucesivamente hasta un cierto número, pero los retardos de mayor duración se consiguen con rutinas ficticias; por ejemplo, la siguiente rutina produce un retardo de 27 estados T:

```
CALL DELAY
RET
```

En este ejemplo, la instrucción CALL emplea 17 estados T, y la instrucción RET emplea 10. El retardo que se obtendrá en un procesador de 2 MHz será de 13,5 microsegundos. Un retardo algo mayor se obtiene incluyendo instrucciones NOP al comienzo de la rutina.

Para retardos mayores hay que servirse de un bucle. En el siguiente ejemplo se carga un registro con el valor que irá decrementándose dentro del bucle. La rutina proporciona un retardo de 99 estados T (49,5 microsegundos a 2 MHz).

INSTRUCCIÓN	TIEMPO (EN ESTADOS T)
CALL DELAY	17
(BUCLE DE RETARDO)	
LD B,5	7
DEC B	4
JR NZ,LOOP	12 (o bien 7, si verdadero)

Las tres instrucciones que comienzan con LD B,5 son el bucle de retardo en sí. Como ocurre con la rutina en código máquina para un 6502, la duración total de esta rutina varía según el valor que se introduce en el registro. El número total de ciclos de reloj que emplea puede ser expresado así:

$$C = 24 + (N \times 16) - 5$$

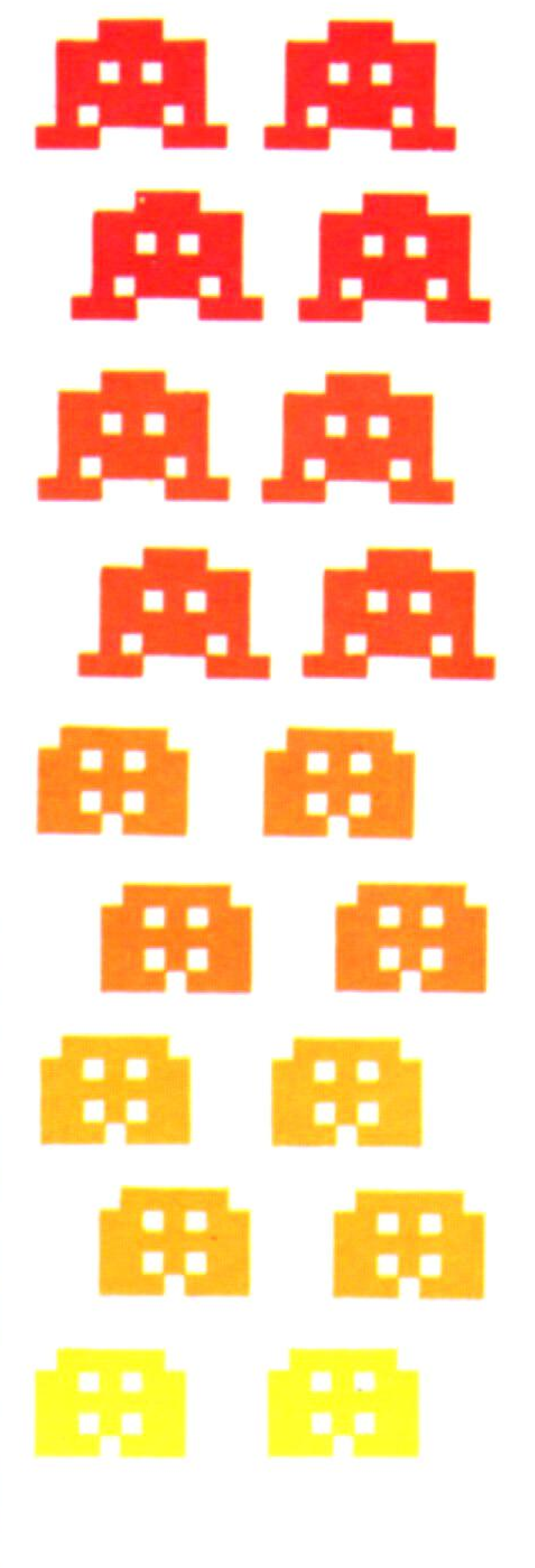
siendo N el valor cargado en el registro B.

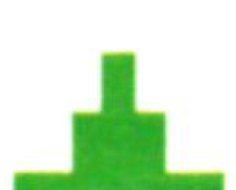
También se pueden utilizar bucles anidados. Pero debemos tener en cuenta otras consideraciones. Primero, que todo registro empleado durante la rutina debe antes "salvarse" *(pushed)* para no perder su contenido. Segundo, que algunas máquinas tienen interrupciones propias de su hardware que pueden alterar el período de retardo. Las interrupciones enmascarables se pueden desactivar y reactivar mediante las instrucciones DI y EI. La siguiente rutina emplea bucles anidados:

INSTRUCCIÓN	FUNCIÓN	TIEMPO (EST. T)
DI	Desactiva interrupciones	4
PUSH DE	Salva el contenido del reg.	11
LD D,nn	Valor del contador interior	7
LD E,nn	Valor del contador exterior	7
CALL OLOOP	Salto al contador exterior	17
POP DE	Restaura el contenido	10
EI	Reactiva interrupciones	4
:		
(OLOOP)		
DEC E	Decremento del bucle exterior	4
RET Z	Final si es cero	11 o 5
(ILOOP)		
DEC D	Decremento del bucle interior	4
JP Z,OLOOP	Bifurca si D es cero	10
JP ILOOP	Si no, continúa bucle interior	10

En esta rutina el retardo aumenta si aumenta el valor del registro E. La rutina acabará al decrementar el registro E y obtener un valor de cero. Observe que si el bucle interior llega hasta cero y el exterior todavía tiene una valor mayor que uno, el bucle interior será inicializado a 255 e irá contando hasta cero antes de pasar el control al bucle exterior.

Invasión cronometrada
Los retardos en código máquina son necesarios en programas de juegos, y en particular cuando hay un objeto que se mueve en la pantalla y ha de ser manipulado por el jugador. Un ejemplo ya clásico es el juego de los *Invasores del espacio*. Sin retardos cronometrados el movimiento de los extraterrestres invasores sería excesivamente rápido. Mientras que gracias a los retardos cuidadosamente cronometrados el movimiento se controla según sea necesario para que el juego discurra debidamente





Conexiones a todo color

Prism es una empresa en constante expansión desde que se iniciara como distribuidora de los productos Sinclair



El presidente de la empresa
Richard Heath, presidente de Prism, que la fundó como empresa subsidiaria de ECC Publications

Mientras la mayoría de las empresas de la industria del ordenador personal se contentan con una perspectiva a corto plazo del mercado, satisfaciendo la demanda inmediata de hardware y software, Prism piensa en el futuro. Una de las más importantes distribuidoras de hardware, Prism jugó un papel fundamental en el desarrollo del Micronet (la primera base de datos a gran escala que se puso a disposición de los usuarios de máquinas personales) y en la actualidad ha tomado a su cargo la distribución de robots de bajo coste.

La empresa la creó ECC Publications en 1982 para desarrollar el Micronet, bajo la dirección de Richard Heath y Bob Denton. El Micronet utiliza el Prestel, uno de los mayores sistemas públicos de videotexto, para posibilitar a los usuarios de una amplia gama de ordenadores personales cargar software, acceder a información e intercambiar "correo electrónico" (véase p. 581). ECC Publications ya había lanzado la revista *Sinclair User*, aunque en aquel entonces los productos Sinclair sólo se vendían por correspondencia o a través de la cadena minorista W. H. Smith. *Sinclair User* alcanzó un enorme éxito, a pesar del escepticismo inicial de Terry Cartwright, actual director de marketing de Prism. "Yo creía que Sinclair no era más que flor de un día —admite—, pero acudimos a la primera Feria del Micro ZX con 8 000 solicitudes de suscripción y las repartimos todas en unas horas."

Sinclair decidió introducirse en el mercado minorista de las tiendas comerciales céntricas y Prism firmó a tiempo un contrato para distribuir el ZX81 y el Spectrum, recientemente lanzado. De hecho, el nombre de la empresa se eligió de forma deliberada para inducir en la mente del público una asociación con el Spectrum; al fin y al cabo, si uno dirige un haz de luz a través de un prisma, ¡acaba con un espectro de colores! Recientemente Prism ha declarado haber vendido más de 500 000 máquinas Sinclair: alrededor del 25 % de todas las ventas de ordenadores personales realizadas en Gran Bretaña.

En marzo de 1983 Prism lanzó el Micronet, en sociedad con la British Telecom y Telemap. Prism se encargó del hardware, distribuyendo una gama de modems (fabricados por O E Ltd y Thorn EMI) para las máquinas más populares. La contribución más reciente a esta gama fue un modem para utilizar con el Commodore 64.

En la actualidad el Micronet tiene alrededor de 10 000 suscriptores, pero recientemente Prism ha vendido su participación en la red y ahora se está concentrando en comercializar y distribuir hardware para ordenadores. Además del Spectrum, en la actualidad Prism se encarga de comercializar las máquinas Oric y Atmos, y la máquina de oficina portátil Wren ("reyezuelo", en castellano), "marca de la casa".

Prism también se está introduciendo en un área nueva: la distribución de robots personales. Éste es un campo que está despertando un creciente interés y ahora Prism comercializa el "Topo", un robot importado de Estados Unidos, así como diversos kits de robot económicos que se venden bajo el nombre comercial "Movits".

Terry Cartwright considera que los robots son un área de enorme expansión. "Existe un inmenso interés por los robots —afirma—. No sé lo que la gente hará con ellos, pero en 1976 nadie sabía tampoco lo que sucedería con los ordenadores Apple." Asimismo, la empresa pretende empezar la distribución del Sinclair QL a finales de este año. Cartwright espera que continúe la diversificación de Prism en el futuro. "La expansión a nivel exterior será la principal prioridad para los próximos doce meses", afirma.



Bonos de suscripción
La empresa tiene planeado alquilarles estos modems a los clientes, junto con una suscripción de un año al Prestel y al Micronet

Pájaro madrugador
El Wren (reyezuelo), la máquina de oficina portátil de Prism, basada en el Z80, viene equipado con dos unidades de disco y un modem incorporado para conectarlo al Prestel

Editorial Delta, S.A.



10048
9 788485 822836